ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИИ

КАФЕДРА ИНФОРМАЦИОННЫХ И РОБОТОТЕХНИЧЕСКИХ СИСТЕМ

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТА КОМПЬЮТЕРНОЙ ТЕХНИКИ И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В ИНСТИТУТЕ ИиЦТ «НИУ БЕЛГУ»

Выпускная квалификационная работа
обучающегося по направлению подготовки
09.03.02 Информационные системы и технологии
очной формы обучения, группы 12001509
Мухторова Усмона Давлаталиевича

Научный руководитель
к.э.н., ст. преподаватель,
Нестерова Е.В.

БЕЛГОРОД 2019

СОДЕРЖАНИЕ

РЕФЕРАТ

Разработка информационной система учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ» – Мухторов Усмон Давлаталиевич, выпускная квалификационная работа бакалавра, Белгород, Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), количество страниц 51, включая приложения 62, количество рисунков 33, количество таблиц 8, количество использованных источников 24.

КЛЮЧЕВЫЕ СЛОВА: информационная система, база данных, учет компьютерной техники, приложение.

ОБЪЕКТ ИССЛЕДОВАНИЯ: процесс учета компьютерной техники и программного обеспечения.

ПРЕДМЕТОМ ИССЛЕДОВАНИЯ: автоматизация учета компьютерной техники и программного обеспечения.

ЦЕЛЬ РАБОТЫ: совершенствование процесса учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ».

ЗАДАЧИ ИССЛЕДОВАНИЯ: анализ и описание предметной области, разработка информационных моделей учета компьютерной техники и программного обеспечения «Как есть» и «Как будет»; проектирование информационного обеспечения системы и структуры базы данных; разработка интерфейса программы; программная реализация ИС учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ».

ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ: проведен анализ и описание предметной области; разработано техническое задание; разработаны модели для проектирование; программно реализованы модели и разработана ИС учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ».

ВВЕДЕНИЕ

В настоящее время в связи со сложной структурой образования ВУЗов приходится сталкиваться с задачей учета компьютерной техники, программного обеспечения, относящиеся к внутренним подразделениям ВУЗа, а именно института. При этом возникают проблемы перед администрацией института, когда, собранные в бухгалтерской системе данные не спасают от вирусной атаки компьютеры, довольно много времени уходит на сбор заявок на замену техники, приходится вести учет параметров, по которым техника должна попадать в программу модернизации, следует формировать перечень техники на замену автоматически из бухгалтерской учетной системы и сформировать быстрый статистический отчет.

В настоящее время решение этих проблем решается вручную с помощью офисных средствами для учета компьютерной техники, например, вести таблицу в MS Excel, или другом табличном редакторе. При этом необходимость ввода данных вручную влечет повышение трудоёмкости при обновлении и времени на проверку изменений в конфигурации оборудования.

Следующий способ решение проблемы – использовать программный продукт для автоматического сбора информации об аппаратной и программной конфигурации компьютеров с последующим занесением в базу данных. Такой вариант имеет следующие преимущества: сбор конфигурации производится автоматически и требует лишь небольшого редактирования в дальнейшем. Хранение информации в базе данных значительно облегчает последующее построение отчетов. Автоматическое обновление информации о конфигурациях компьютеров позволяет повысить оперативность при работе с базой данных.

Объектом исследования является изучение процесса учета компьютерной техники и программного обеспечения в институте инженерных и цифровых технологий (ИИиЦТ) НИУ БелГУ.

Предмет исследования является автоматизация учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ».

Целью выпускной квалификационной работой (ВКР) является совершенствование процесса учета компьютерной техники и программного обеспечения в институте ИиЦТ технологий НИУ “БелГУ” за счет разработки информационной системы.

Для достижения поставленной цели выпускной квалификационной работы необходимо решить следующие задачи:

– анализ и описание приметной области;

– разработка технического задания;

– разработка моделей для проектирования;

– программная реализация разработанных моделей;

– описание разработанной информационной системы.

Актуальность разработки такой информационной системы вызван тем, что в процессе функционирования отдела, решается перечень рутинных задач, при выполнении которых формируется множество документов долговременного хранения, оперативная работа с которыми возможна только в рамках автоматизированной информационной системы.

В первом разделе выпускной квалификационной работы описаны анализ и описание предметной области.

Во втором разделе было описано проектирование системы и разработка технического задания.

В третьем разделе описана программная реализация, создание базы данных разработка и тестирование информационной системы

Данная работа состоит из 51 страницы, 33 рисунка, 8 таблиц, 24 литературных источников и 1 приложений.

1 Анализ и описание предметной области

В данном разделе выпускной квалификационной работы предоставлен анализ и исследование предметной области. Автор описывает предметную область, выявляет необходимость в разработке информационной системы (ИС), далее предоставляются выбранные средства разработки ИС и обоснование их выбора.

Анализ предметной области является одной из важной задачей, с которой сталкивается разработчик, перед тем как приступить к проектированию и разработки информационной системы (ИС).

На данном этапе требуется изучить, осмыслить и провести анализ над предметной областью. Анализ предметной области позволит выявить сущности, определить первоначальные требования к функциональности информационной системы и определить её границы.

1.1 Описание предметной области

Организационные структуры являются одним из элементов, которые необходимо учитывать при управлении проектом, поскольку это фактор, который может существенно повлиять на доступность ресурсов и оказать решающее влияние на способ управления проектами.

Структура университета отображена на рисунке 1.

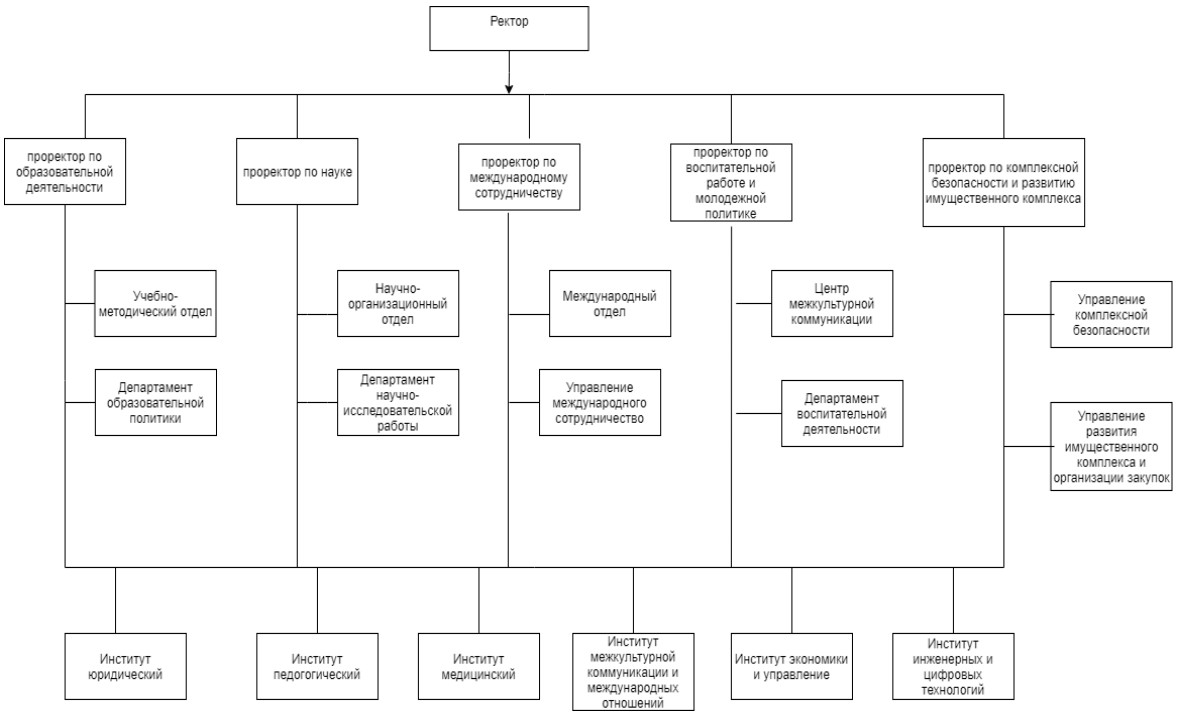


Рисунок 1 - Организационная структура университета НИУ «БелГУ»

Простая структура, в которой процесс принятия решений централизован, а связь между руководителями среднего звена и оперативным уровнем является формальной. Недостатки: организация может быть жесткой и негибкой, что может привести к замедлению работы из-за нехватки времени у руководителей или владельцев для принятия каждого из решений для эффективного выполнения работы.

Институт инженерных и цифровых технологий создан на основании приказа ректора НИУ «БелГУ» от 24.07.2018 № 675-ОД. Обучение студентов осуществляется в соответствии с европейскими стандартами непрерывного трехуровневого высшего образования: бакалавриат, магистратура, аспирантура. Основные усилия коллектива института направлены на подготовку высококвалифицированных кадров в области технических и физико-математических наук[1].

Для практической подготовки обучающихся, в институте предусмотрены компьютерные классы.

Для обслуживания данных классов ведётся учёт, как компьютерной техники, так и программного обеспечения к каждому помещению

(компьютерному классу) привязан ответственный который и производит данный учет.

На данный момент учет компьютерной техники и программного обеспечения в институте является не автоматизированным. Учет ведется с использованием таких программных средств, как MS Excel и MS Word.

MS Excel выступает как хранилища данных, информации о компьютерной техники и ПО, их месторасположения и другое. MS Word используется с целью предоставления отчётов.

Такой учет является сложным и трудоемким для его проведения сотрудниками, работающими в отделе информатизации. Для решения выявленной проблемы было решено разработать ИС которая совершенствует этот процесс.

Разработанная информационная система включит в себя всю необходимую информацию и возможности по ведению учета. Она позволит:

а) создавать, обновлять и удалять информацию о корпусах и аудиториях, в которых располагается компьютерная техника;

б) создавать, обновлять и удалять информацию о компьютерной техники, и установленных на них, программных обсечениях;

в) выводить необходимые отчеты по проделанной работе и всего учета.

Администрирование информационной системы буду проводить сотрудники, отвечающие за компьютерные классы и компьютерную технику которые в них расположены. Для этой возможности в системе предусмотрено создание пользователей, администрирующие систему [2].

1.2 Выбор средств разработки информационной системы

Выбор средств разработки для реализации программного обеспечения включает в себя определение, какая конкретная система будет выбрана для

управления базой данных и с помощью какого языка будет построен функционал информационной системы.

В современном мире существует много различных СУБД, такие как: Oracle, Firebird, SQLite, MySQL, PostgreSQL, mSQL, Sybase ASE и др.

Рассмотрим наиболее распространенные СУБД такие как: MySQL,SQLite и PostgreSQL.

SQLite - это библиотека, написанная на языке C, которая реализует автономную систему управления базами данных SQL, без сервера и без конфигурации. Код SQLite находится в открытом доступе и является бесплатным для любого использования, как коммерческого, так и частного. В настоящее время он используется во многих приложениях, в том числе в проектах высокого уровня[3].

Достоинства SQLite:

– файловая структура - вся база данных состоит из одного файла, поэтому её очень легко переносить на разные машины;

– используемые стандарты - хотя может показаться, что эта СУБД примитивная, но она использует SQL. Некоторые особенности опущенны (RIGHT OUTER JOIN или FOR EACH STATEMENT), но основные все-таки поддерживаются;

– отличная при разработке и тестировании - в процессе разработки приложений часто появляется необходимость масштабирования. SQLite предлагает всё, что необходимо для этих целей, так как состоит всего из одного файла и библиотеки написанной на языке C.

Недостатки SQLite:

– отсутствие системы пользователей - более крупные СУБД включают в свой состав системы управления правами доступа пользователей. Обычно применения этой функции не так критично, так как эта СУБД используется в небольших приложениях;

– отсутствие возможности увеличения производительности - опять, исходя из проектирования, довольно сложно выжать что-то более производительное из этой СУБД.

MySQL - это имя системы, которая позволяет управлять базами данных. Это наиболее часто используемая опция для приложений, основанных на интернете. MySQL Этот администратор реляционных баз данных был создан компанией MySQL AB. Поэтому база данных MySQL распространяется по-разному. Существует версия с общедоступной лицензией GNU (так называемая Community Edition) и другая, ориентированная на компании, предоставляющие дополнительные услуги и продукты. Следует отметить, что Facebook, YouTube и Twitter, среди других самых посещаемых сайтов в мире, работают с MySQL. GNU / Linux, Mac OS X, SunOS, Solaris и различные версии Windows (Windows 7, Windows 10, Windows Vista и другие), среди многих других платформ, позволяют работать с MySQL. Важно отметить, что как администратор реляционной базы данных MySQL предлагает гибкость и скорость, поскольку данные хранятся в отдельных таблицах, которые связаны друг с другом, а не в одном большом файле. По оценкам, в настоящее время работает более 6 миллионов копий MySQL, что свидетельствует об огромной популярности этой системы[4].

Достоинства MySQL:

– простота в работе - установить MySQL очень просто. Дополнительные приложения, позволяет довольно легко работать с БД;

– богатый функционал - MySQL поддерживает большинство функционала SQL;

– безопасность - большое количество функций обеспечивающих безопасность, которые поддерживается по умолчанию;

– масштабируемость - MySQL легко работает с большими объемами данных и легко масштабируется;

– скорость - упрощение некоторых стандартов позволяет MySQL значительно увеличить производительность.

Недостатки MySQL

– известные ограничения - по задумке в MySQL заложены некоторые ограничения функционала, которые иногда необходимы в особо требовательных приложениях;

– проблемы с надежностью - из-за некоторых способов обработки данных MySQL (связи, транзакции, аудиты) иногда уступает другим СУБД по надежности;

– медленная разработка - Хотя MySQL технически открытое ПО, существуют жалобы на процесс разработки. Стоит заметить, что существуют другие довольно успешные СУБД, созданные на базе MySQL, например MariaDB.

PostgreSQL является самым профессиональным из всех трех рассмотренных нами СУБД. PostgreSQL - это объектно-ориентированная система управления реляционными базами данных с открытым исходным кодом, опубликованная по лицензии PostgreSQL, 1 аналогичная BSD или MIT.

Как и многие другие проекты с открытым исходным кодом, разработка PostgreSQL не управляется компанией или человеком, а направляется сообществом разработчиков, которые работают в самоотверженных, альтруистических, бесплатных или поддерживаемых коммерческих организациях. Это сообщество называется PGDG (PostgreSQL Global Development Group)[5].

В PostgreSQL нет диспетчера дефектов, поэтому очень сложно узнать состояние его дефектов.

Достоинства PostgreSQL:

– открытое ПО соответствующее стандарту SQL - PostgreSQL - бесплатное ПО с открытым исходным кодом. Эта СУБД является очень мощной системой;

– большое сообщество - существует довольно большое сообщество в котором вы запросто найдёте ответы на свои вопросы;

– большое количество дополнений - несмотря на огромное количество встроенных функций, существует очень много дополнений, позволяющих разрабатывать данные для этой СУБД и управлять ими;

– расширения - существует возможность расширения функционала за счет сохранения своих процедур;

– объектность - PostrgreSQL это не только реляционная СУБД, но также и объектно-ориентированная с поддержкой наследования и много другого.

Недостатки PostgreSQL:

– производительность - при простых операциях чтения PostgreSQL может существенно замедлить сервер и быть медленнее своих конкурентов, таких как MySQL;

– популярность - по своей природе, популярностью эта СУБД похвастаться не может, хотя и присутствует довольно большое сообщество;

– хостинг - в силу выше перечисленных факторов иногда довольно сложно найти хостинг с поддержкой этой СУБД.

На основе просмотра общих достоинств и недостатков для разработки информационной системы учета компьютерной техники и программного обеспечения был выбран СУБД PostgreSQL.

Существует большое количество языков программирования. Каждый язык был придуман и создан для решения определенного типа задач. Большая часть языков пересекается в функционале, поэтому одну и ту же задачу можно решать различными инструментами. Но при этом у каждого языка программирования есть свои преимущества и недостатки, даже при решении одного типа задач. [6]

Рассмотрим три самых популярных языков программирование.

C# - объектно-ориентированный язык программирования. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java[7].

Достоинства:

– имеет статическую типизацию;

– поддерживает полиморфизм;

– поддерживает перегрузку операторов;

– поддерживает комментарии в формате XML.

Недостатки:

– не имеет доступа к машинному коду;

– не работает без framework.

C++ – компилируемый статически типизированный язык программирования общего назначения. C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений (игр). Существует множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ[8].

Достоинства:

– поддерживает процедурное программирование;

– поддерживает объектно-ориентированное программирование;

– поддерживает обобщённое программирование;

– обеспечивает модульность;

– обеспечивает раздельную компиляцию;

– обеспечивает обработку исключений;

– обеспечивает абстракцию данных;

– обеспечивает объявление типов (классов) объектов;

– обеспечивает виртуальные функции.

Недостатки:

– отсутствие сборщика мусора;

– невозможность компиляции шаблонов;

– сложный в освоении.

Java. Язык программирования, изначально предназначенный для описания и разработки объектных моделей. Может использоваться любой сервер. Лицензия - свободная. На Java можно успешно реализовывать проекты любого масштаба. [9]

Достоинства:

– гибкая система безопасности;

– независимость байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина;

– модульность;

– масштабируемость.

Недостатки:

– исполнение байт-кода виртуальной машиной снижает производительность программ и алгоритмов, реализованных на языке Java;

– чрезмерная нагрузка, система пользователя должна обладать достаточно мощными вычислительными ресурсами, особенно при работе с приложениями, содержащими сложные пользовательские интерфейсы.

На основе просмотра общих достоинств и недостатков для разработки информационной системы учета компьютерной техники и программного обеспечения был выбран язык программирование C#.

Выводы по первому разделу

В данном разделе выпускной квалификационной работы, был проведен анализ и описание предметной области; был сделан выбор средств разработки информационной системы учета компьютерной техники и программного обеспечения, что является решением одних из основных задач работы.

Постановка следующих задач для проектирования и реализация разрабатываемой информационной системы, выстраивается следующим образом: разработка технического задания; разработка модели для проектирования; программная реализация разработанных моделей; описание разработанной ИС.

2 Разработка информационной системы учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ»

Под разработкой информационной системы для института ИиЦТ «БелГУ» подразумевается достижение следующей цели, а это совершенствование процесса учета путем разработки информационной системы.

В ее функции входят:

– учета компьютерной техники и программного обеспечения;

– составление отчета.

Для администрирования информационной системы требуется разработать отдельную форму содержимым, в процессе которой будут производиться такие операции как обеспечение безотказной работы системы и его ресурсов, управление доступом (назначение прав), а также обеспечение доступности и безопасности.

В выпускной квалификационной работе необходимо разработать ИС учета компьютерной техники и программного обеспечения.

2.1 Техническое задание на разработку информационной системы учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ»

Информационная система учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ» предназначена для повышения эффективности учета.

Основанием для разработки информационной системы является необходимость совершенствование процесса учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ».

2.1.1 Назначение разработки

Под информационной системой учета компьютерной техники и программного обеспечение подразумевается решение следующих задач:

а) авторизация пользователя;

б) хранение информации о количестве компьютеров и программное обеспечения;

в) хранение информации об аудиториях и корпусах;

г) добавление, удаление и изменение компьютеров, и программное обеспечения;

д) учет компьютеров и программного обеспечения;

е) добавление новых пользователей;

ж) выход из системы.

2.1.2 Требования к ИС учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ БелГУ

Система должна обеспечивать следующие функции:

а) ввод, вывод, редактирование, хранение информации о компьютерах:

- мониторы;
- процессор;
- видеокарта;
- оперативная память;
- материнская плата;
- операционная система;

б) ввод, вывод, редактирование, хранение информации об ответственных лицах:

- фамилия;
- имя;
- отчество;

- год рождения;
- номер телефона;
- должность.

в) ввод, вывод, редактирование, хранение информации о корпусах:

- номер корпуса;
- адрес корпуса.

г) ввод, вывод, редактирование, хранение информации об аудиториях:

- номер аудитории;
- ответственные за аудиторию;

д) ввод, вывод, редактирование, хранение о программном обеспечении:

- наименование программы;
- производитель;
- версия.

Входной информацией системы является:

а) информация о количестве компьютерах;

б) информация о программном продукте;

в) логины и пароли администратора;

г) учет компьютеров и программного обеспечения.

Выходной информацией системы является:

а) управляющие воздействия на базу данных;

б) учет компьютерной техники и программной обеспечения;

в) отчет.

По требованиям к надежности система должна:

- проводить контроль вводимой информации;
- блокировать некорректные действия пользователя при работе с системой;
- обеспечивать целостность данных;
- обеспечивать права доступа;
- обеспечивать защиту персональных данных.

В связи с тем, что использовать систему будут пользователи средней и низкой квалификации, то интерфейс системы должен быть простым и интуитивно понятным. Ввод информации должен осуществляться в наиболее унифицированных формах.

Требования к составу и параметрам технических средств: настоящая система должна работать на процессорах совместимых с процессором IBM; оперативная память на каждой ЭВМ не менее 128 Мб; наличие доступа к локальной сети и сети Интернет; свободное место на жестком диске не менее 10Мб. Требования к информационной и программной совместимости: система должна работать под управлением ОС Windows; СУБД PostgreSQL; другое ПО выбирается по решению разработчика. Основным критерием является низкая стоимость.

2.1.3 Технико-экономические показатели

Затраты на разработку (ИС) учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ БелГУ сведены в таблицу 1.

Таблица 1 – Смета затрат на разработку информационной системы (ИС)

Элементы затрат	Стоимость, руб.
Компьютер Pentium G5300	30 000
Хозяйственный инвентарь (мебель)	10 000
Амортизация оборудования и инструментальные средства	8 000
Хоз. материалы	1000
Электроэнергия	1000
Накладные расходы	10 000
Продолжение таблицы 1	
Всего	60 000

Стадии и этапы разработки

Общая продолжительность разработки и внедрения системы составляет 4 месяца с 01/03/2018 по 08/06/2018. Общая стоимость работ составляет 60000 руб. График реализации проекта представлен в таблице 2.

Таблица 2 – График реализации проекта

Этапы реализации проекта	Месяцы			
	1	2	3	4
1. Разработка информационной системы учета компьютерной техники и программного обеспечения НИУ БелГУ				
1.1. Техническое задание.				
1.2. Эскизный проект.				
1.3. Технический проект.				
1.4. Рабочий проект.				
1.5. Внедрение.				
2. Покупка ЭВМ, оборудования и инструментальных средств для заказчика (осуществляется за средства заказчика).				
3. Обучение персонала (осуществляется за средства заказчика, согласно отдельному договору).				
4. Эксплуатация				

2.2 Разработка модели использования

В Unified Modeling Language диаграмма вариантов использования - это улучшенная форма диаграммы поведения UML. Унифицированный язык моделирования (UML) определяет графическую нотацию для представления прецедентов, называемую моделью прецедентов. UML не определяет стандарты для письменного формата для описания вариантов использования, и поэтому многие люди не понимают, что эта графическая нотация определяет

природу варианта использования; однако графическая нотация может дать только простой обзор варианта использования или набора вариантов использования. Диаграммы вариантов использования часто путают с вариантами использования. Хотя эти две концепции взаимосвязаны, варианты использования гораздо более подробны, чем диаграммы вариантов использования. В концепциях более одного варианта использования должны быть подробно описаны, чтобы определить, что делает вариант использования [10].

На рисунке 2.1 представлена диаграмма вариантов использования для зарегистрированного пользователя.

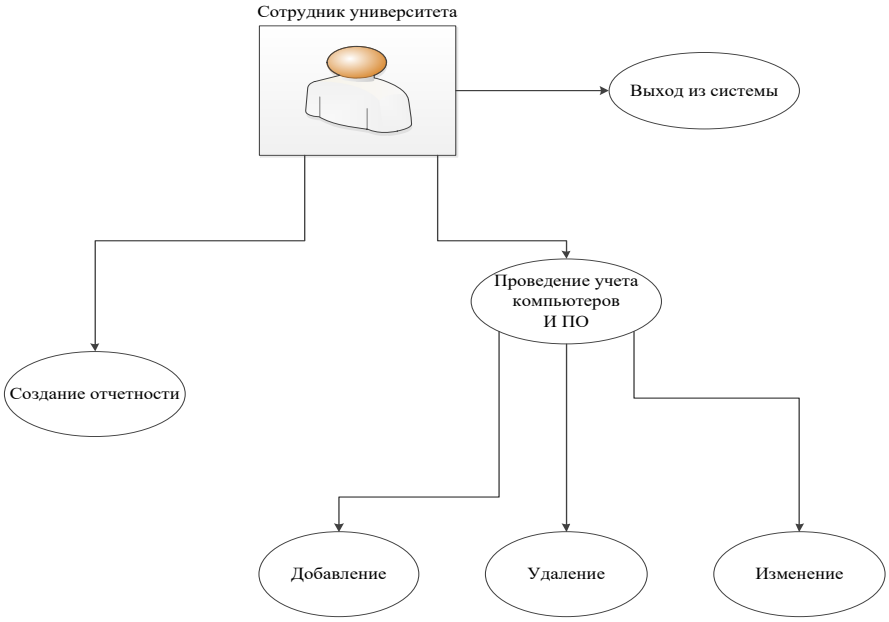


Рисунок 2.1 – Диаграмма вариантов использования

Описание вариантов использования представлено в таблицах3-8.

Таблица 3 – Вариант использования – Изменение

Название	Изменение
Описание	Отображение информации по изменению оборудования
Частота использования	Является используемым
Входные условия	Данные о компьютерах и ПО
Выходные условия	Изменённые данные

Продолжение таблицы 3

Нормальное направление	1 Пользователь запрашивает требующуюся информацию для отображения, изменения, удаления. Вводит новую информацию. 2 Выполняет запрос к БД 3 Отображается, изменяется или удаляется запрашиваемая информация. Добавляется новая информация.
Альтернативные направления	Нет
Исключения	1. Выводится сообщение об ошибке, если введены некорректные данные.

Таблица 4 – Вариант использования – Удаление

Название	Удаление
Описание	Отображение информации по удалению оборудования
Частота использования	Является используемым
Входные условия	Данные о компьютере и ПО
Выходные условия	Удаления данных
Нормальное направление	1 Пользователь запрашивает требующуюся информацию для отображения, изменения, удаления. Вводит новую информацию. 2 Выполняет запрос к БД 3 Отображается, изменяется или удаляется запрашиваемая информация. Добавляется новая информация.
Альтернативные направления	Нет
Исключения	1. Выводится сообщение обошибке,если введены некорректные данные.

Таблица 5 – Вариант использования - Добавление

Название	Добавление
Описание	Отображение информации по добавлению оборудования
Частота использования	Является используемым
Входные условия	Данные о компьютере и ПО
Выходные условия	Добавление новых данных
Нормальное направление	1 Пользователь запрашивает требующуюся информацию для отображения, изменения, удаления. Вводит новую информацию. 2 Выполняет запрос к БД 3 Отображается, изменяется или удаляется запрашиваемая информация. Добавляется новая информация.
Альтернативные направления	Нет
Исключения	1. Выводится сообщение об ошибке, если введены некорректные данные.

Таблица 6 – Вариант использования – Проведение учета компьютеров и ПО

Название	Проведение учета компьютеров и ПО
Описание	Проведение учета компьютеров и ПО
Частота использования	Является часто используемым.
Входные условия	Данные о компьютерах
Выходные условия	Данные по компьютерам., расположение оборудования, количество.
Нормальное направление	1. Пользователь запрашивает информацию по оборудованию. 2. Выполняет запрос к БД 3. Отображается список оборудования их местоположение, характеристики. 4. Добавляется или изменяется информация по имеющемуся оборудованию.
Альтернативные направления	Нет
Исключения	Нет

Таблица 7 – Вариант использования – Создание отчетности

Название	Создание отчетности
Описание	Создание выходной отчетности о проведение учета, добавлении ПО на компьютер, перемещении существующего
Частота использования	Является часто используемым.
Входные условия	Данные о компьютере и ПО и аудиториях
Выходные условия	Созданная отчетность.
Нормальное направление	1 Пользователь запрашивает требующуюся информацию по отчетности для отображения. 2 Выполняет запрос к БД 3 Отображается, запрашиваемая информация.
Альтернативные направления	Нет
Исключения	Нет

Таблица 8 – Вариант использования – Выход из системы

Название	Выход из системы
Описание	Завершение работы с данными информационной системы.
Частота использования	Является часто используемым.
Входные условия	Нет
Выходные условия	Нет
Нормальное направление	1 Пользователь выбирает пункт меню «Выход»; 2 Завершение сеанса пользователя;
Альтернативные направления	Нет
Исключения	Нет

2.3 Проектирование информационной системы

AllFusion® Process Modeler (AllFusion PM) - мощное решение для моделирования, которая может помочь визуализировать, анализировать и улучшать сложные процессы бизнеса. Модель процесса позволяет четко документировать важные аспекты любого бизнес-процесса, например, какие действиями нужно, как они выполняются и контролируются, какие ресурсы необходимы для выполнения и какие выгоды или результаты происходят. Это обеспечивает изображению интегрировать, как ваша компания выполняет задачи, от небольших рабочих процессов ведомственные комплексные организационные функции. [11]

AllFusion совмещает в одном инструменте средства моделирования функций (IDEF0), потоков данных (DFD) и потоков работ (IDEF3).

Контекстная диаграмма учета компьютерной техники и программного обеспечения представляет собой самое общее описание системы и ее взаимодействие с внешней средой. Контекстная диаграмма состоит из одного блока, описывающего функцию верхнего уровня, ее входы, выходы, управления, и механизмы, вместе с формулировками цели модели и точки зрения, с которой строится модель[12].

Конктекстная диаграмма учета компьютеной техники и программного обеспечения приведена на рисунке 2.2.

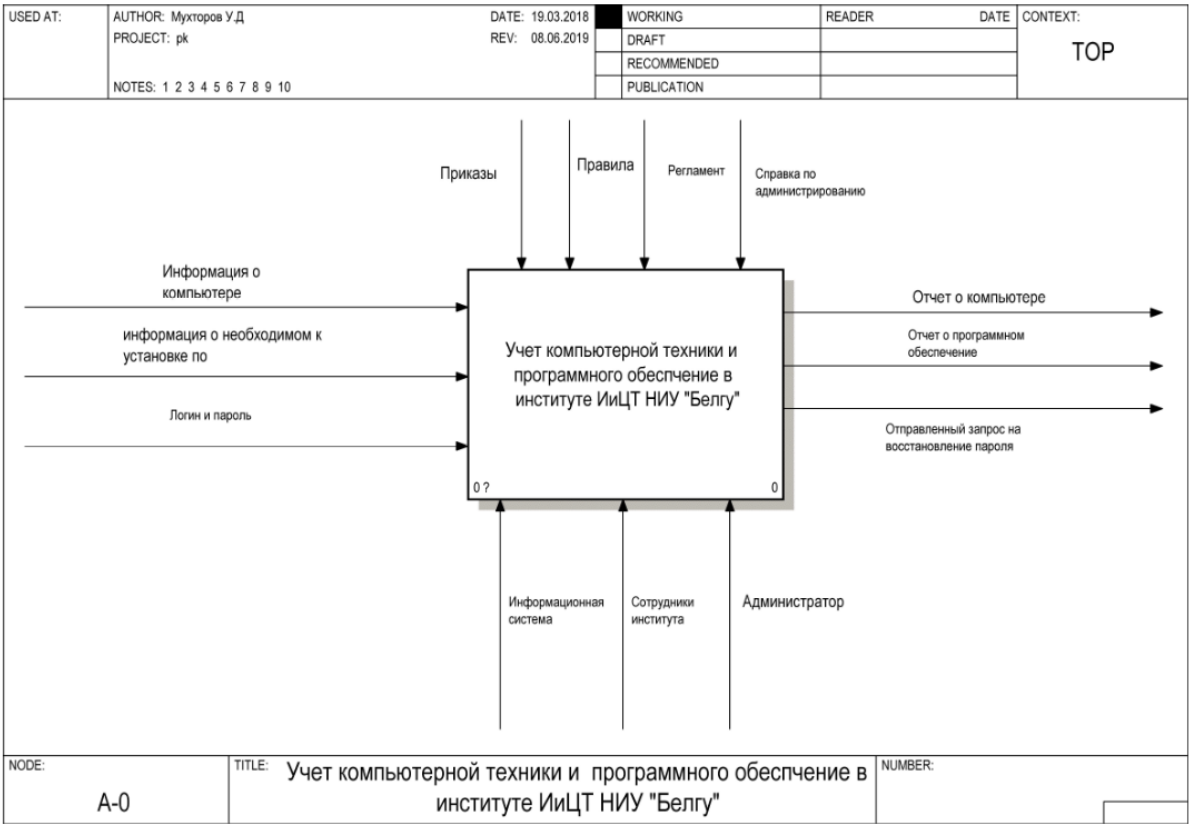


Рисунок 2.2 – Контекстная диаграмма с использованием методологии IDEF0

Входными стрелками в данной композиции являются: логин и пароль пользователя системы, информация о компьютере и информация о необходимом к установке ПО.

Выходными стрелками в данной композиции являются: отчет о компьютерах, отчет о программном обеспечении и отправленный запрос пользователя на восстановление пароля.

Стрелками управления являются: приказы, правила, регламенты и справка по администрированию.

Стрелками механизма являются: администратор, сотрудники университета и информационная система[13].

Декомпозиция данной контекстной диаграммы показана на рисунке 2.3.

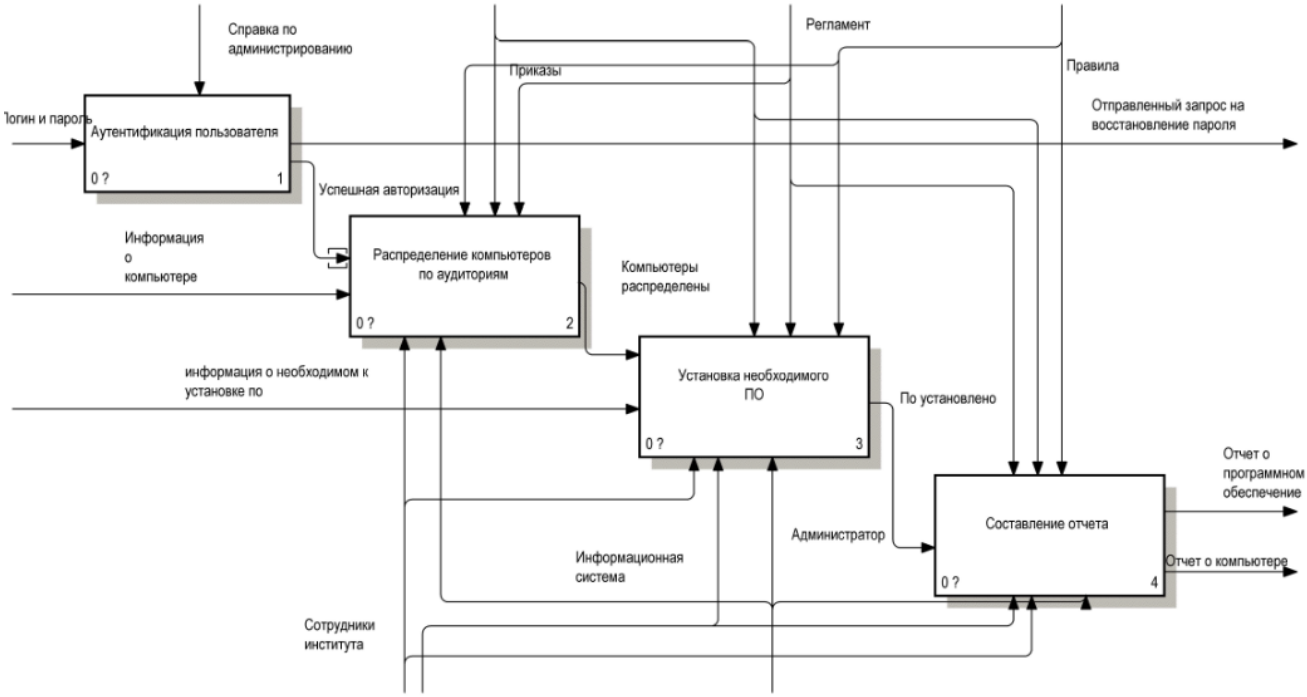


Рисунок 2.3 – Декомпозиция контекстной диаграммы

На рисунке 2.3 предоставлены четыре основных процесса участвующих в использовании и администрировании информационной системы. Данные процессы называются: «Аутентификация пользователя», «Распределение компьютеров по аудиториям», «Установка необходимого ПО» и «Составление отчета».

Ниже предоставлено более детальное представление каждого из процессов. На рисунке 2.4 детально предоставлен процесс «Аутентификация пользователя».

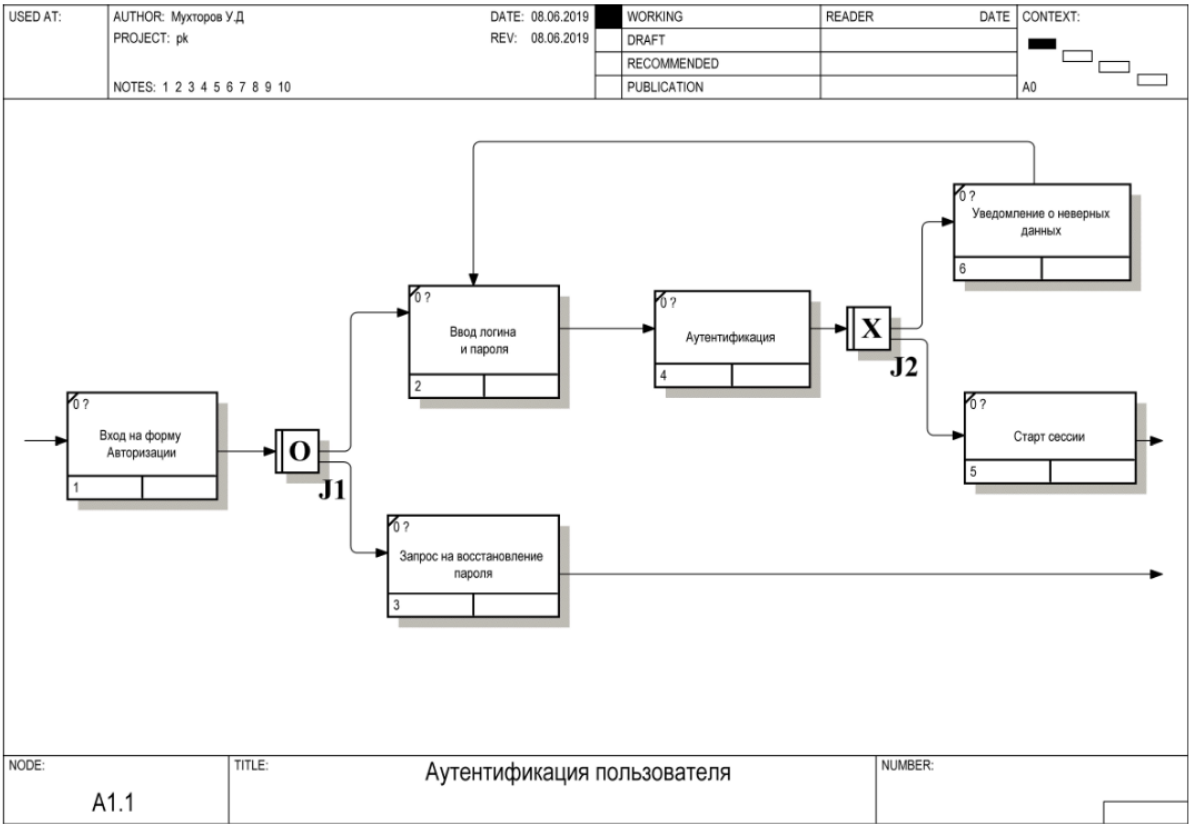


Рисунок 2.4 – Декомпозиция блока "Аутентификация пользователя"

На рисунке 2.4 продемонстрирован один из важных процессов описывающий авторизацию пользователя, перед тем как он войдет в информационную систему.

Данный процесс разбит на 6 блоков, каждый из них является операцией в зависимости, от которого может происходить определенный сценарий.

Начальной операцией является вход пользователя на форму авторизации, после этого пользователь может выбрать одно из двух возможных действий, первое это ввести логин и пароль, второе это отправить запрос на восстановление пароля, если пользователь забыл пароль или потерял свои данные.

После ввода логина и пароля идет аутентификация пользователя, на программном уровне, где проверяется, существует ли такой пользователь в базе данных. В итоге процесса аутентификации может произойти только один из двух сценариев. Успешная аутентификация дает старт процессу «Старт

сессии», где запоминается время входа пользователя в систему, а проваленная аутентификация дает старт процессу «Уведомление о неверных данных», после которой пользователю требуется опять провести операцию «Ввод логина и пароля».

На рисунке 2.5 представлена декомпозиция блока «Распределение компьютеров по аудиториям».

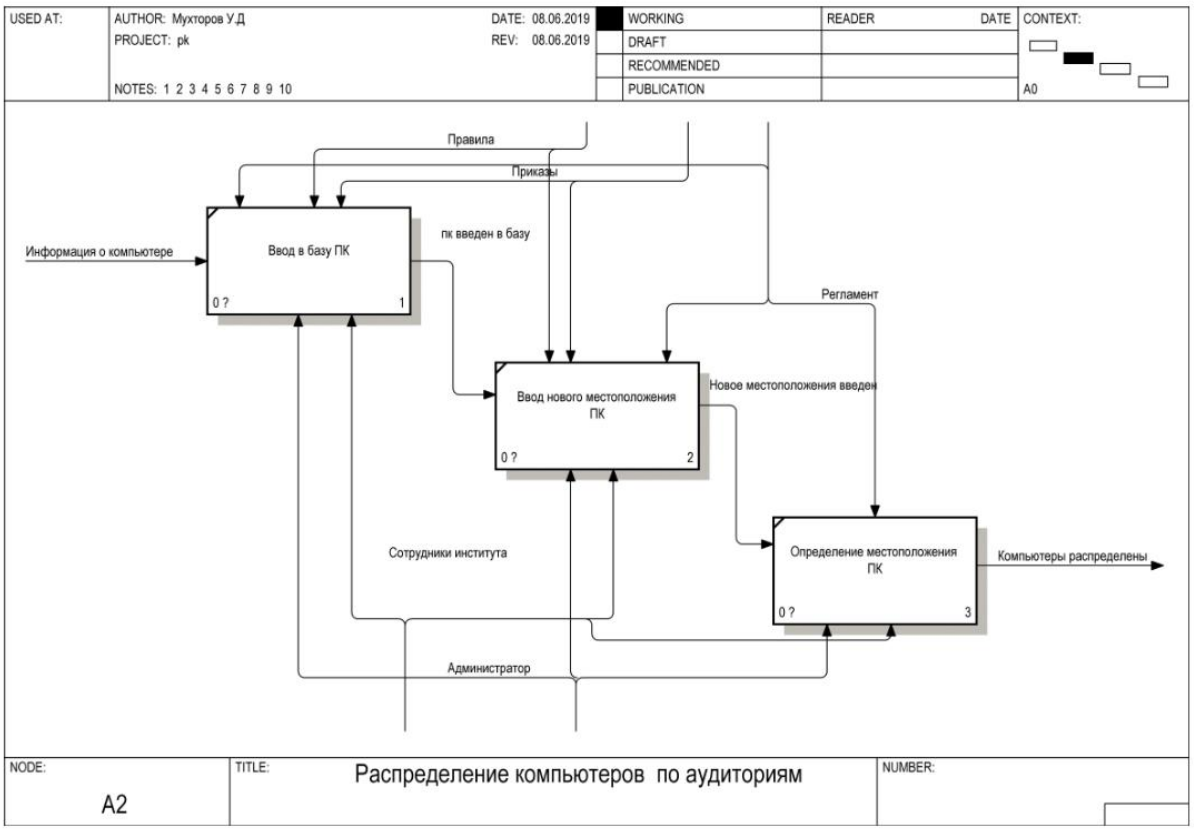


Рисунок 2.5 – Декомпозиция блока "Распределение компьютеров по аудиториям"

На рисунке 2.5 продемонстрирован еще один из важных процессов описывающий информацию о компьютерах их местоположении. Данный процесс разбит на такие структурные элементы, как: «Ввод в базу ПК», «Ввод нового местоположения ПК» и «Определения местоположения ПК».

Суть этих элементов заключается в том, пользователь вводит данные о компьютере и о его новом местоположении, потом определяет новое местоположение компьютера.

Детализация блока «Установка необходимого ПО» представлена на рисунке 2.6

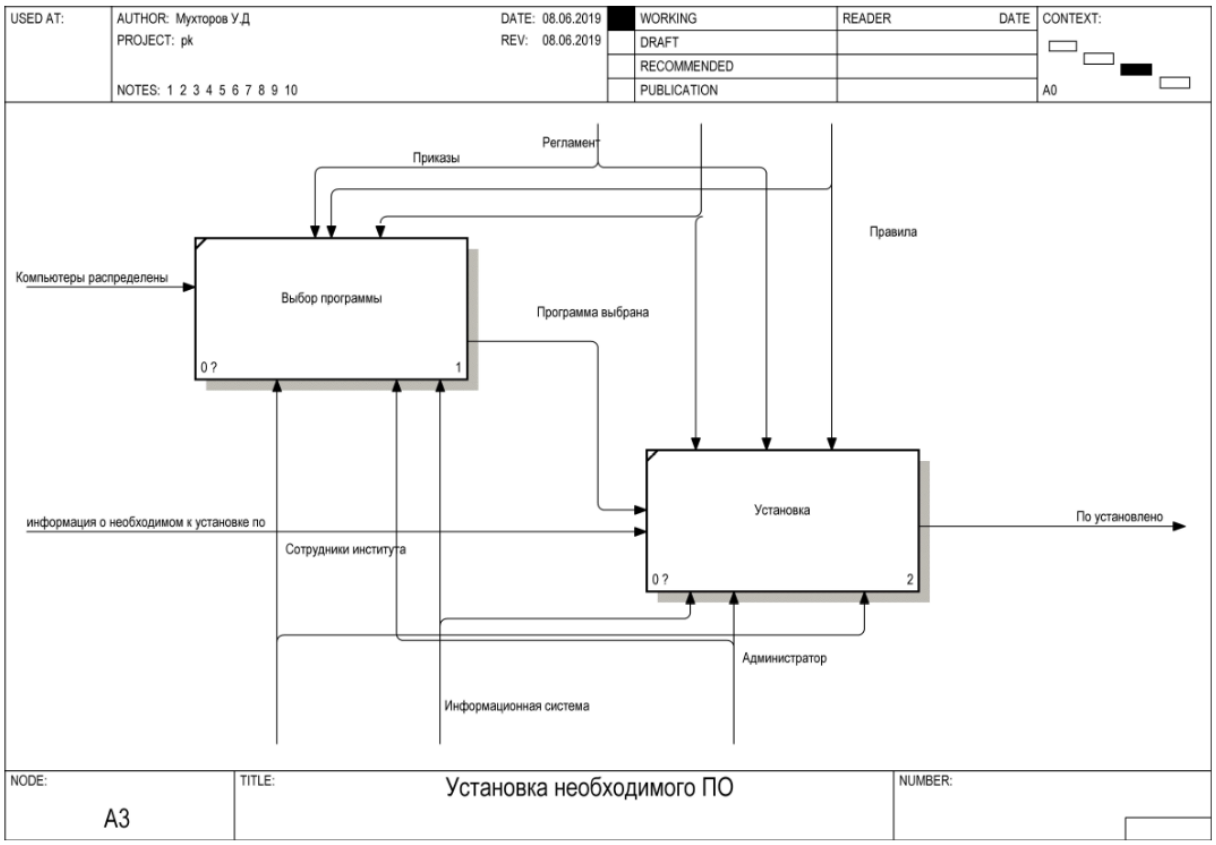


Рисунок 2.6 – Декомпозиция блока " Установка необходимого ПО "

На рисунке 2.6 предоставлен процесс, описывающий возможность установки программ на компьютер. Данный процесс разбит на такие структурные элементы, как: «Выбор программы» и «Установка».

Первым из процессов является «Выбор программы», пользователь выбирает программу после выбора программы непосредственно идет процесс «Установка», при помощи данного процесса пользователь устанавливает программу на компьютер.

Далее на рисунке 2.7 детально продемонстрировано процесс «Составление отчета».

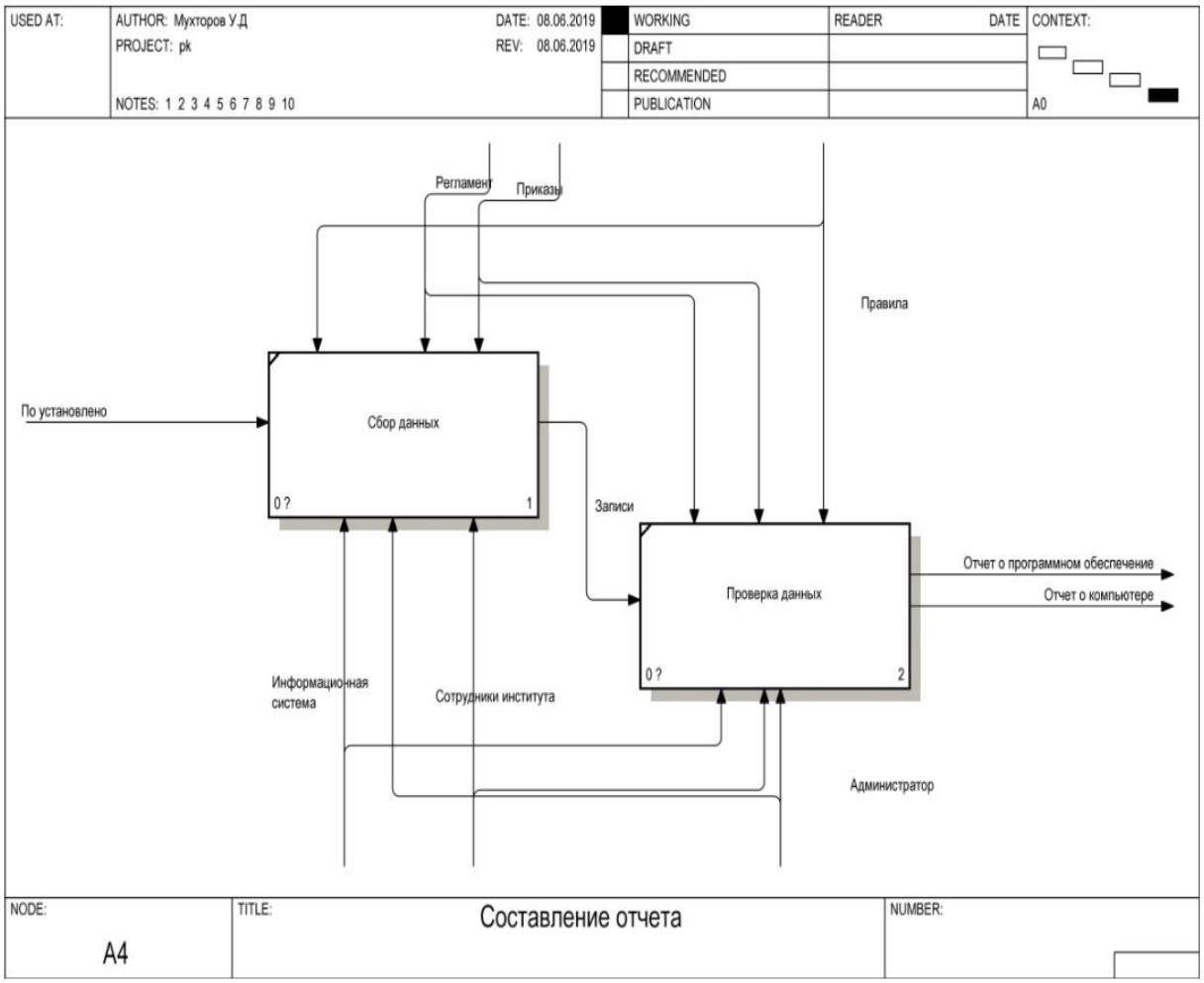


Рисунок 2.7 – Декомпозиция блока " Составление отчета "

На рисунке 2.7 продемонстрирован процесс, описывающий возможность составление отчета. Данный процесс разбит на такие структурные элементы, как: «Сбор данных» и «Проверка данных».

Первым из процессов является «Сбор данных», после установки ПО пользователь проверяет записи, потом идет процесс «Проверка данных» и на выходе готовые отчеты.

В итоге была смоделирована информационная система учета компьютерной техники и программного обеспечения.

Выводы по второму разделу

Таким образом, можно подвести следующие выводы, что в данном разделе выпускной квалификационной работы было достигнуто решение одних из самых главных поставленных задач, а это:

а) было разработано техническое задание проекта;

б) была разработана и спроектирована модели;

Решение данных задач позволяет перейти к программной реализации разработанных моделей и описанию информационной системы учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ».

3 Программная реализация информационной системы

3.1 Инфологическое проектирование

Базу данных можно проектировать любым способом в зависимости от нескольких факторов. Важнейшим фактором является совместимость используемой вами системы управления базами данных с конкретной моделью. Большинство систем управления базами данных разрабатываются с учетом конкретной модели данных и требуют от пользователей принятия этой модели, хотя некоторые из них совместимы с несколькими моделями. Кроме того, разные модели применяются к разным этапам процесса проектирования базы данных. Концептуальные модели данных высокого уровня лучше подходят для создания карт связей между данными так, как люди воспринимают эти данные. С другой стороны, логические модели, основанные на записях, более точно отражают способы хранения данных на сервере. Выбор модели данных также зависит от того, согласовываете ли вы свои приоритеты с сильными сторонами базы данных конкретной модели, включают ли эти приоритеты скорость, снижение затрат, удобство использования или что-то еще. [14]

Концептуальная схема - это высокоуровневое описание структуры базы данных независимо от СУБД, которая будет использоваться для ее манипулирования. Концептуальная модель - это язык, который используется для описания концептуальных схем. Целью концептуального проекта является описание информационного содержимого базы данных, а не структур хранения, которые будут необходимы для обработки этой информации. Логическое проектирование начинается с концептуальной схемы и в результате дает логическую схему[15].

Для информационной системы учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ» выделены следующие сущности:

- корпус;
- аудитория;
- лицензия;
- ответственный;
- программное обеспечения;
- компьютер;
- учет;

На основании приведенных выше сущностей, построена модель схемы отношений с помощью среды Erwin Data Modeler [8], которая представлена на рисунке 2.7

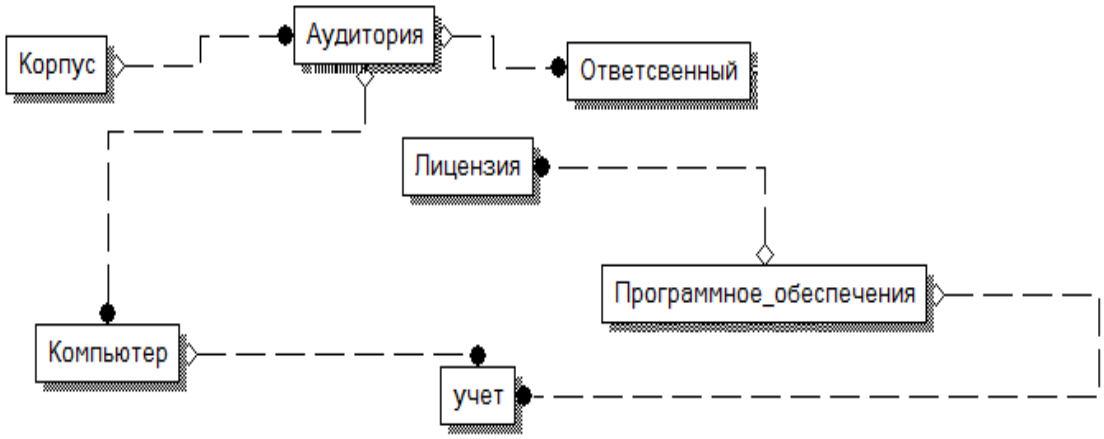


Рисунок 2.7 – Модель схемы отношений

Связь работает путем сопоставления данных в ключевых столбцах; обычно это столбцы с одним и тем же именем в обеих таблицах. Существует три типа связей между таблицами. Тип создаваемой связи зависит от того, как определены связанные столбцы.

Связь «один ко многим» (1 – n) самая распространенная. В этом случае запись таблицы А может быть связана с несколькими таблицами В. Это наиболее распространенный и используемый тип, и существует множество случаев; например, адреса с людьми, которые живут в нем, название компании с ее работниками, поставщиков с продуктами, которые обслуживают [16].

На схеме, представленной на рисунке 2.7, такая связь существует между следующими таблицами:

- «корпус» и «аудитория»: один корпус может иметь несколько аудитории, но одна аудитория не может быть в нескольких корпусах;
- «аудитория» и «ответственный»: одно ответственной лицо может отвечать за несколькими аудиториям, но несколько ответственных лиц не могут отвечать за одну аудиторию;
- «аудитория» и «компьютер»: каждая аудитория может иметь несколько компьютеров, но один компьютер не может быть в нескольких аудиториях;
- «программное обеспечения» и «лицензия»: каждая программа может иметь несколько лицензии, но одно лицензия не может быть в нескольких программах;

В связи «многие ко многим» (n–n) если несколько записей А могут быть связаны с несколькими из В и наоборот. Классический пример - иметь два стола, один для актеров, а другой для фильмов, поскольку каждый актер обычно работал над несколькими фильмами, и они формируются несколькими актерами. Таких связей в схеме, представленной на рисунке 2.7 нет[17].

В связи «один к одному появляется, когда запись в таблице А может быть связана только с 1 записью в таблице В. Эта модель появляется в исключительных отношениях, таких как страны-флаги, поскольку каждая страна имеет один официальный флаг, и каждый флаг может быть только, принадлежать стране; другим примером могут быть автомобильные номерные знаки и номер шасси . Такие связи в схеме, представленной на рисунке 2.7 нет[18].

3.2 Даталогическая модель базы данных

Логическая модель данных - это модель, которая не является специфичной для базы данных и описывает аспекты, связанные с потребностями организации в сборе данных и взаимосвязями между этими аспектами[19].

Логическая модель содержит в себе представления объектов атрибутов, отношений, уникальных идентификаторов, подтипов и супертипов и ограничений между отношениями. Логическая модель также может содержать объекты модели предметной области или ссылаться на одну или несколько моделей предметной области или глоссария. Как только отношения и логические объекты определены в логической модели данных, используйте рабочую область для преобразования логической модели в конкретное физическое представление базы данных в форме физической модели данных.

Через рабочую область вы можете создать логическую модель данных из шаблона. Вы также можете импортировать простые типы данных из файла определения схемы XML (.xsd) в логическую модель данных, такую как типы доменов.

Физическая модель данных - это модель базы данных, которая представляет реляционные объекты данных и их взаимосвязи. Физическая модель данных может использоваться для генерации операторов DDL, которые затем могут быть развернуты на сервере базы данных[20].

На рисунке 2.8 приведен пример даталогической модели данных.

Для построения инфологической и даталогической модели был использован язык ER-диаграмм.

В базе данных учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ» будет 7 сущностей:

а) «Корпус» – содержит данные о корпусах;

б) «Аудитория» – содержит информацию об аудиториях;

в) «Ответственный» – содержит информацию об ответственных лицах за аудиторию;

г) «Компьютер» – содержит данные о компьютерах;

д) «Программное обеспечения» – содержит данные о программном продукте;

е) «Лицензия» – содержит данные о лицензиях программ;

ж) «Учет» – содержит данные об учете компьютеров и программных обеспечении

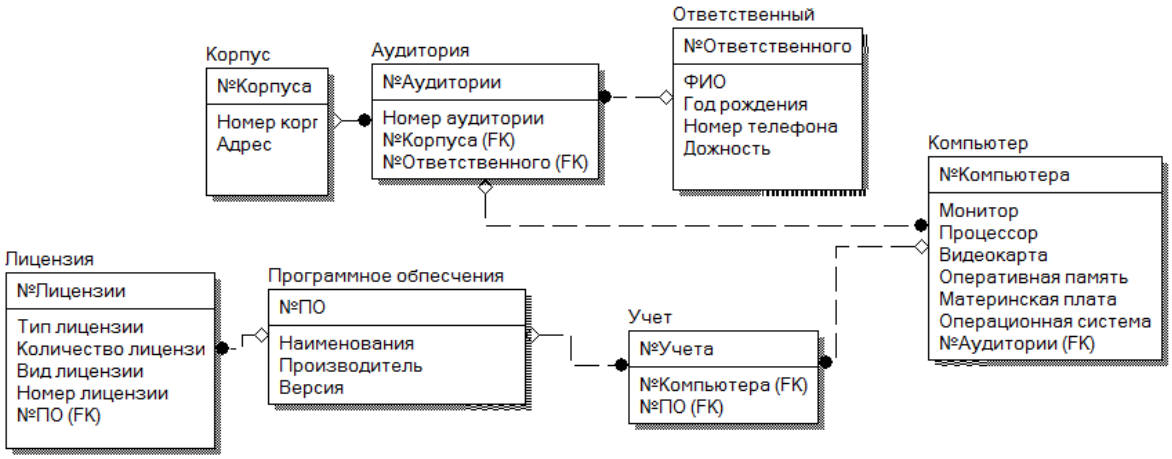


Рисунок 2.8 – Даталогическая модель

Эти сущности и атрибуты являются основой для создания таблиц в базе данных для дальнейшего хранения информации. Изменение данных в одной таблице повлечет за собой изменения в другой, таким образом, обеспечивается целостность данных – еще одна функциональная возможность программы ERwin[21].

Для создания таблиц было создано всего семь сущностей, поскольку этого достаточно для обеспечения полноценной работы разрабатываемой системы.

Таблица «auditoriya» (рисунок 2.9) – это таблица «Аудитории», которая используется для хранения данных об аудиториях.

Содержит следующие поля:

- «id_aud» – код аудитории;

- «nomer_aud» – номер аудитории;
- «id_kor» – связь с таблицей «korpus» корпус;
- «id_ot» – связь с таблицей «otvetstvenniy» ответственный.

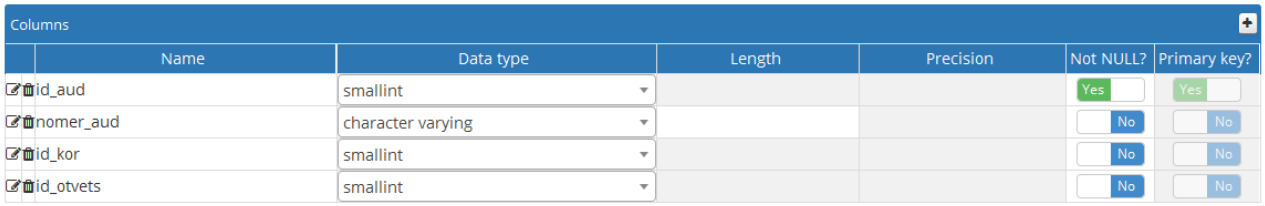


Рисунок 2.9 – Таблица «Аудитория»

Таблица «computer» (рисунок 2.10) – это таблица «Компьютер», в котором хранятся данные о компьютерах.

Содержит следующие поля:

- «id_pk» – номер компьютера;
- «monitor» – монитор;
- «proc» – процессор;
- «videokarta» – видеокарта;
- «ram» – оперативная память;
- «materinskaya_plata» – материнская плата;
- «os» – операционная система;
- «id_aud» – связь с таблицей «auditoriya» аудитория.

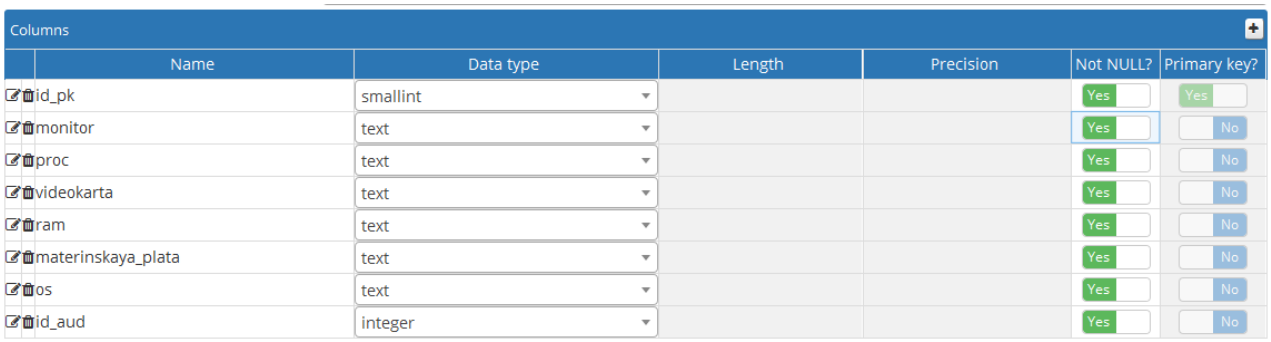


Рисунок 2.10 – Таблица «Компьютеры»

Таблица «korpus» (рисунок 2.11) – это таблица «Корпус», в котором хранится данные о корпусах.

Содержит следующие поля:

- «id_kor» – код корпуса;
- «nomer_kor» – номер корпуса;
- «adress» – адрес корпуса.

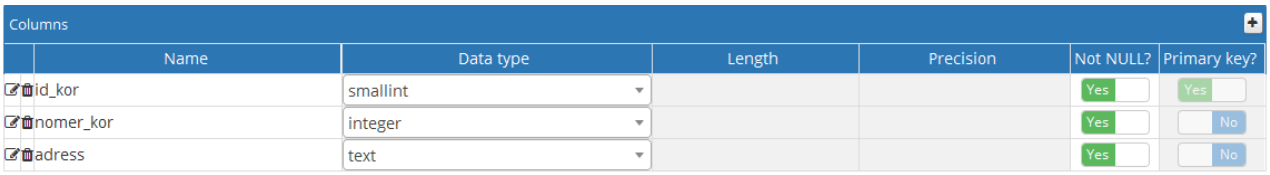


Рисунок 2.11 – Таблица «Корпус»

Таблица «license» (рисунок 2.12) – это таблица «Лицензия», в котором хранится информация о лицензиях программных продукт.

Содержит следующие поля:

- «id» – номер лицензии;
- «type» – тип лицензии;
- «kolichestvo» – количество лицензии;
- «vid_lic» – вид лицензии;
- «number» – номер лицензии;
- «id_po» – связь с таблицей «po» программное обеспечения.

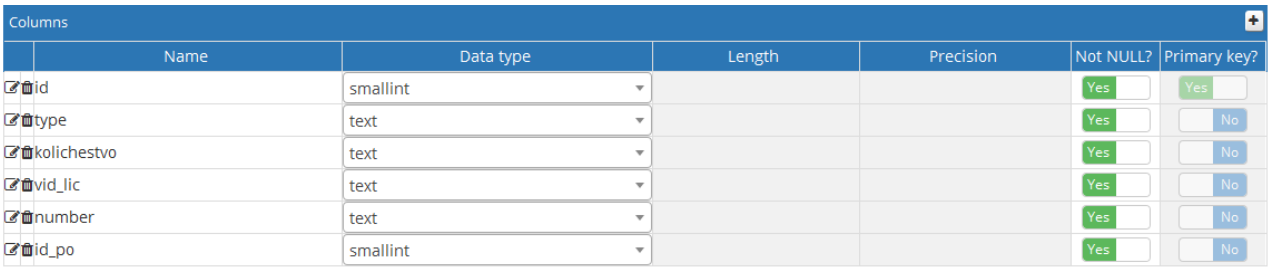


Рисунок 2.12 – Таблица «Лицензия»

Таблица «otvetstevnniy» (рисунок 2.12) – это таблица «Ответственные», в котором хранится данные об ответственных лицах.

Содержит следующие поля:

- «id_ot» –номер ответственного;
- «fio» – ФИО;
- «god_rojdeniya» – год рождения;
- «nomer_telephona» – номер телефона;
- «dolzhnost» – должность.

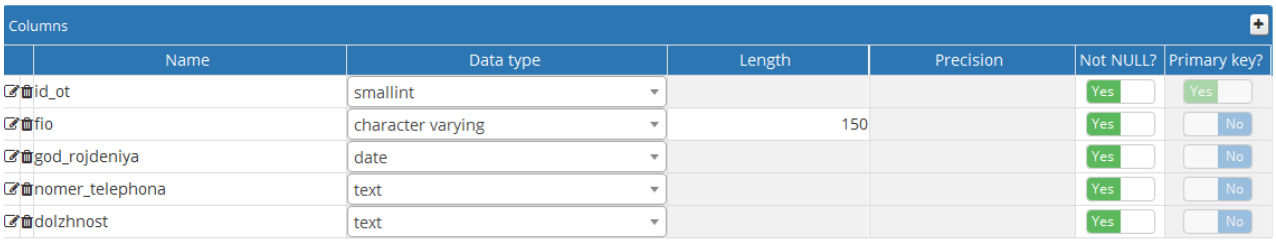


Рисунок 2.13 – Таблица «Ответственный»

Таблица «po» (рисунок 2.15) – это таблица «PO», в котором хранится данные о программах.

Содержит следующие поля:

- «id_po» – номер программного обеспечения;
- «name_po» – наименование программного обеспечения;
- «proizvoditel» – производитель программного продукта;
- «version» – версия.

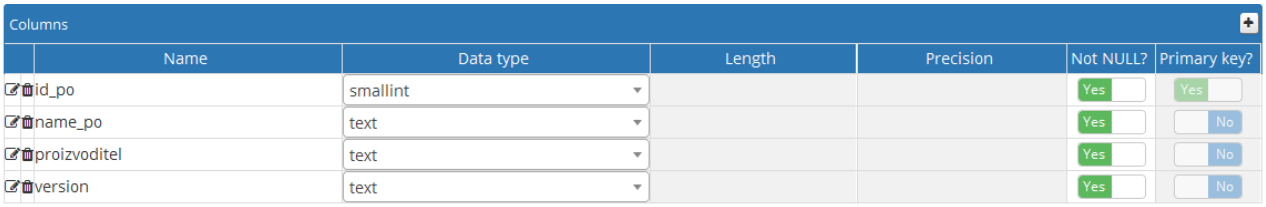


Рисунок 2.15 – Таблица «ПО»

Таблица «uchet» (рисунок 2.16) – это таблица «Учета», в котором хранится данные об учете ПК и ПО.

Содержит следующие поля:

- «id_uchet» – номер учета;
- «id_pk» – связь с таблицей «computer» компьютер;
- «id_po» – связь с таблицей «po» программное обеспечения.

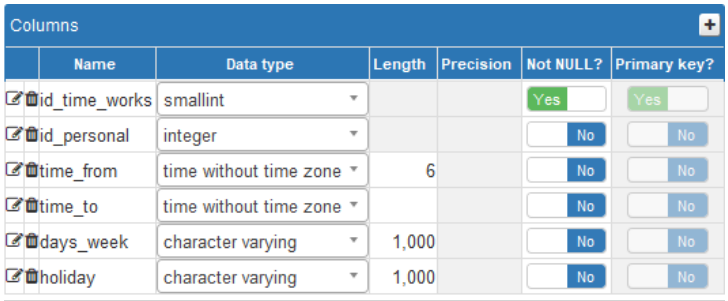


Рисунок 2.16 – Таблица «График работы»

Для разработки базы данных был выбран СУБД – PostgreSQL. Работа с pgAdmin4 продемонстрировано на рисунке 2.17[22].

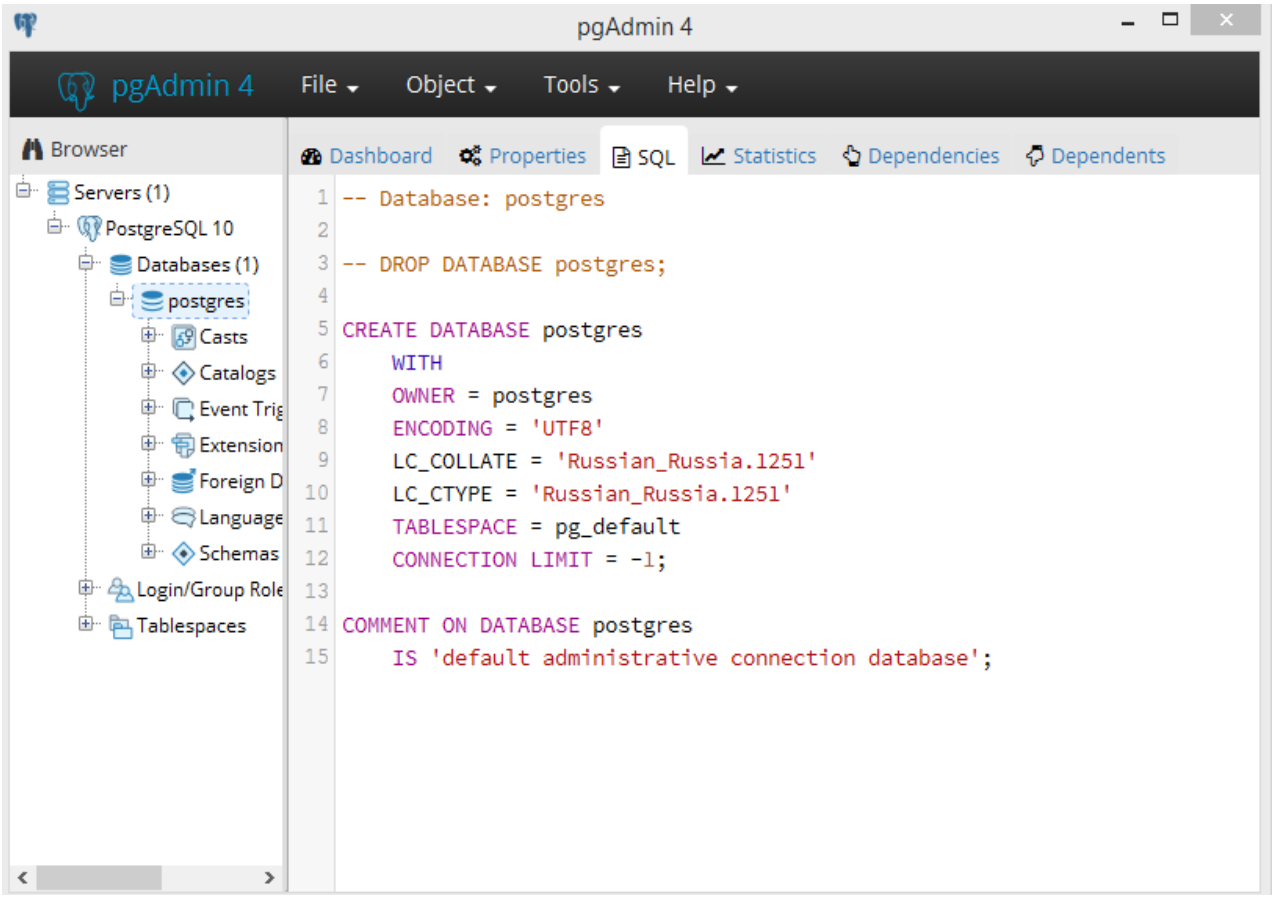


Рисунок 2.17 – Работа с pgAdmin4

С помощью pgAdmin удобно и просто администрировать сервера PostgreSQL, а также создавать базы данных [23].

3.3 Описание разработанной информационной системы

В качестве средства разработки программы используется язык программирования C# и среда разработки Visual Studio 2015[24].

При запуске приложения появляется стартовое окно, где пользователь должен авторизоваться на рисунок 3 это будет продемонстрировано.

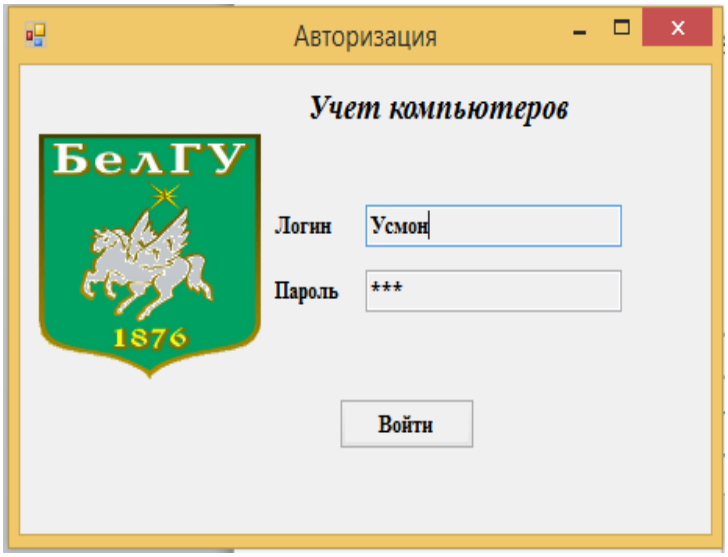


Рисунок 3 – Авторизация пользователя

После авторизации логина и пароля для входа в систему открывается главное окно приложения (рисунок 3.1).

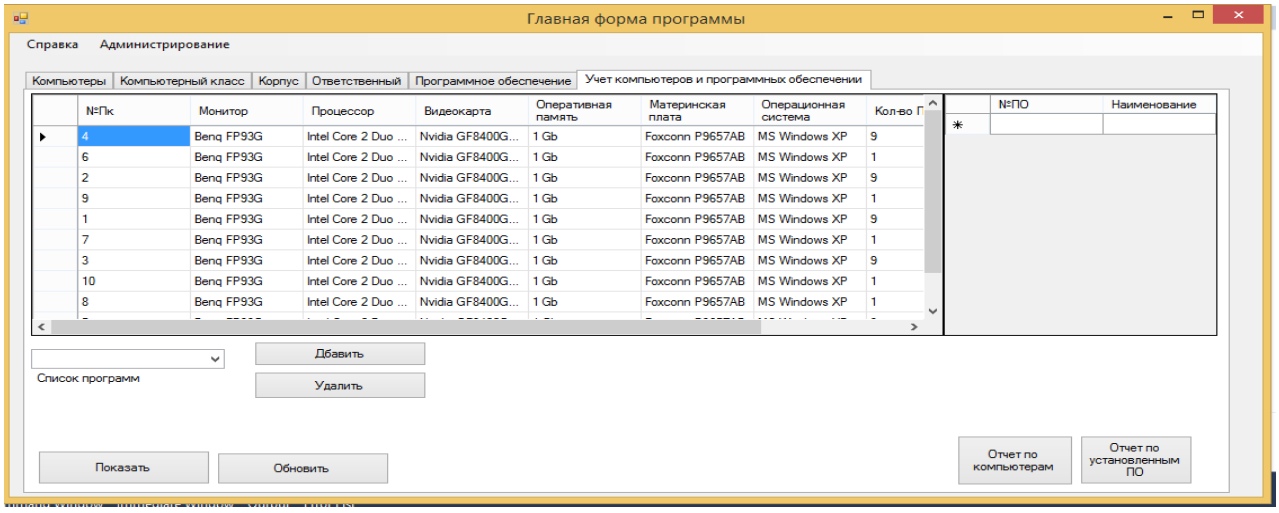


Рисунок 3.1 – Главное окно программы

При открытии главного окна, можно просмотреть какие таблица находятся в приложении, просмотрим несколько из этих таблиц.

На рисунке 3.2 продемонстрировано таблица компьютеры в нем хранятся данные о компьютерах.

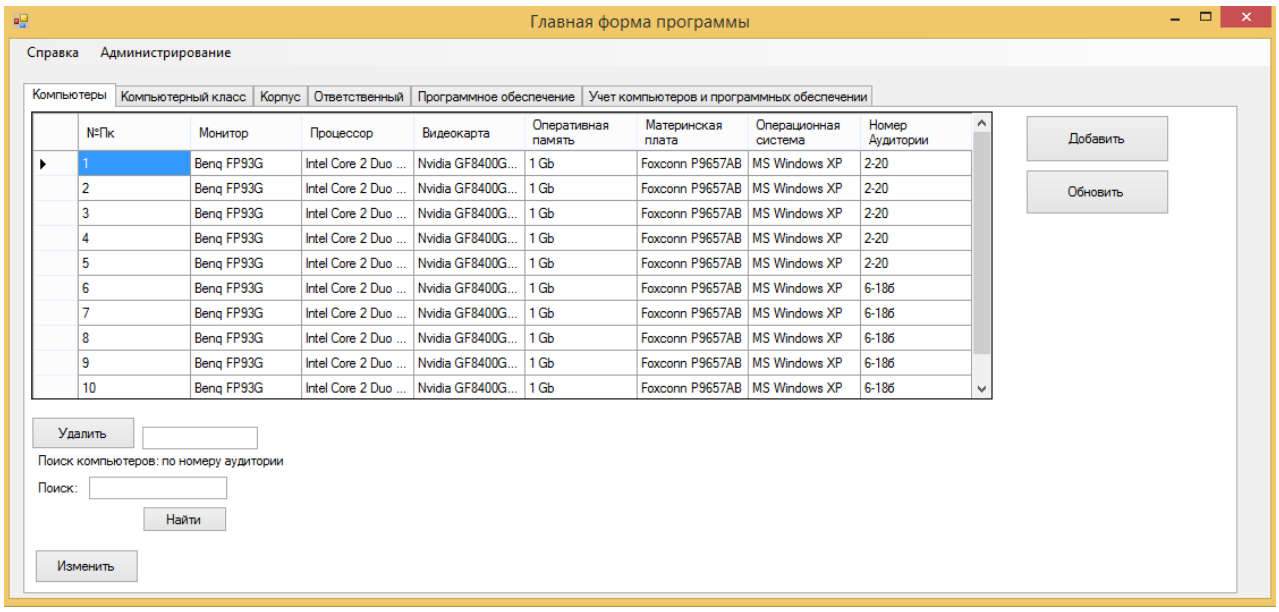


Рисунок 3.2 – Таблица «Компьютеры»

В таблице компьютеры и во всех других таблицах есть такие функции как: удаление, поиск, добавление, изменение и обновление данных.

Воспользуемся функцией добавление нового компьютера, чтобы следующим этапом сделать учет, по этому компьютеру установив в нем программное обеспечения (рисунок 3.3).

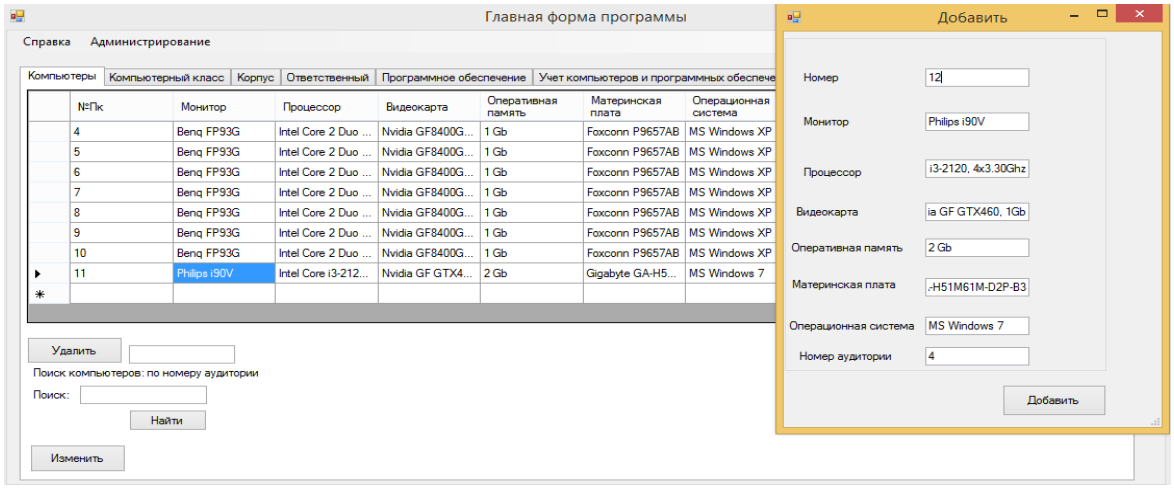


Рисунок 3.3 – Добавление компьютера

После добавление нового компьютера пользователь спокойно может найти этот компьютер по номеру аудитории (рисунок 3.4).

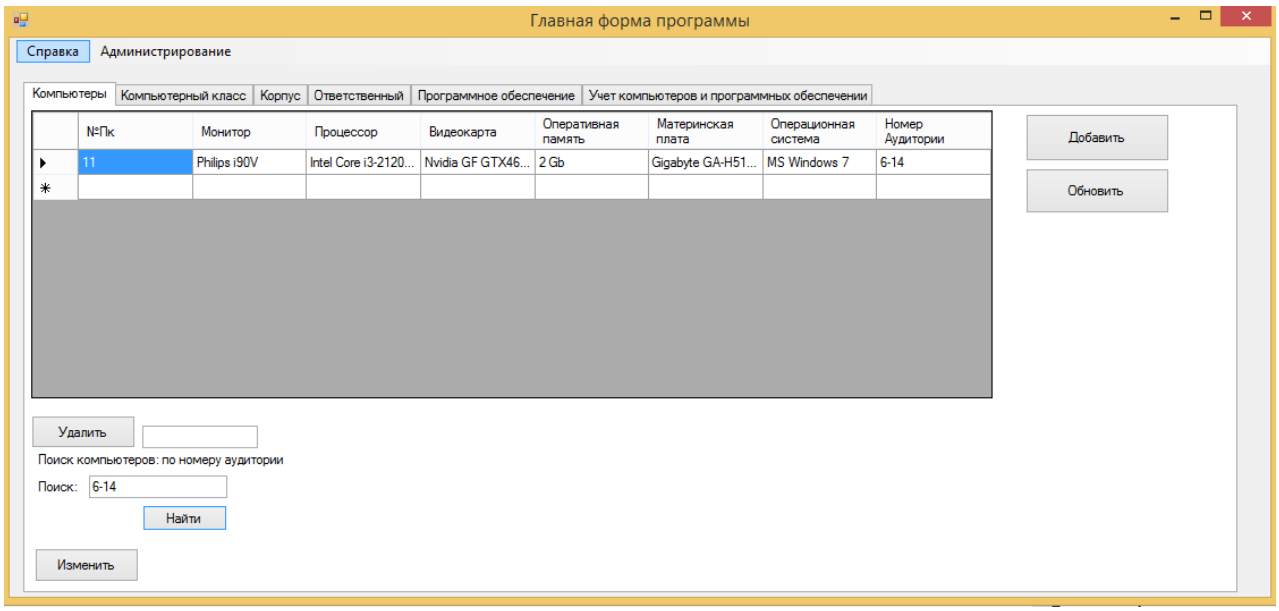


Рисунок 3.4 – Поиск компьютера по номеру аудитории

Далее рассмотрим таблицу «Компьютерный класс» в нем находятся информация о номере компьютерного класса, номере корпуса в котором находится компьютерный класс, и ФИО ответственного лица который отвечает за компьютерный класс (рисунок 3.5).

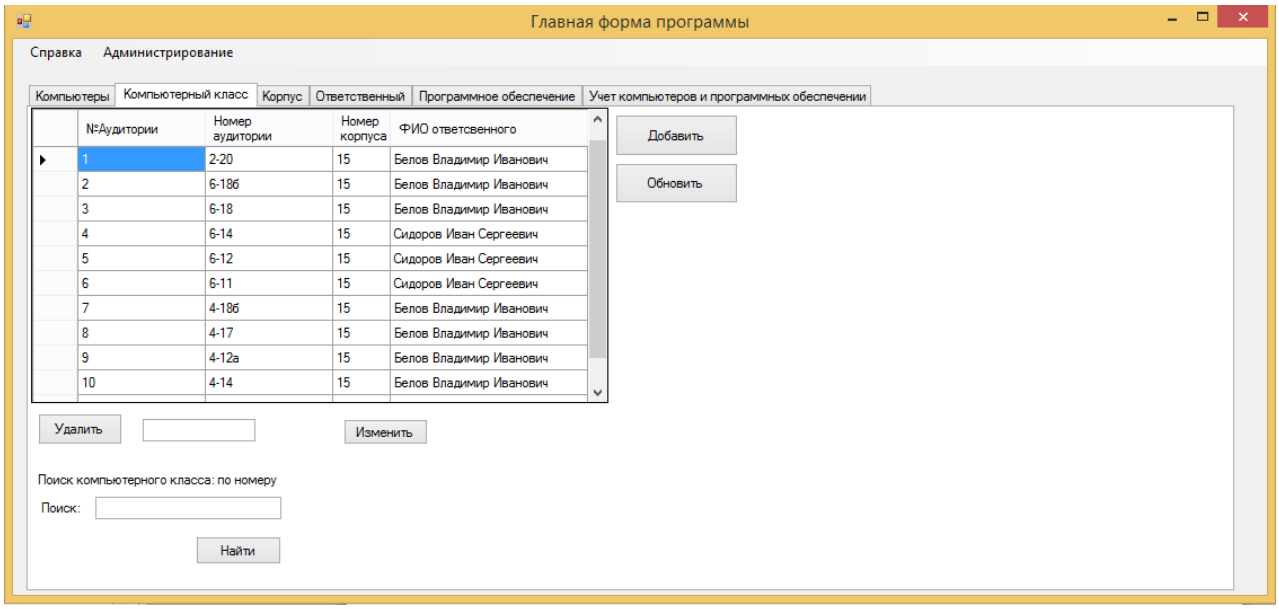


Рисунок 3.5 – Таблица «Компьютерный классы»

После просмотра таблица «Компьютерный класс», переходим на таблицу «Программное обеспечение», в этой таблице хранятся данные о программном продукте и лицензиях (рисунок 3.6).

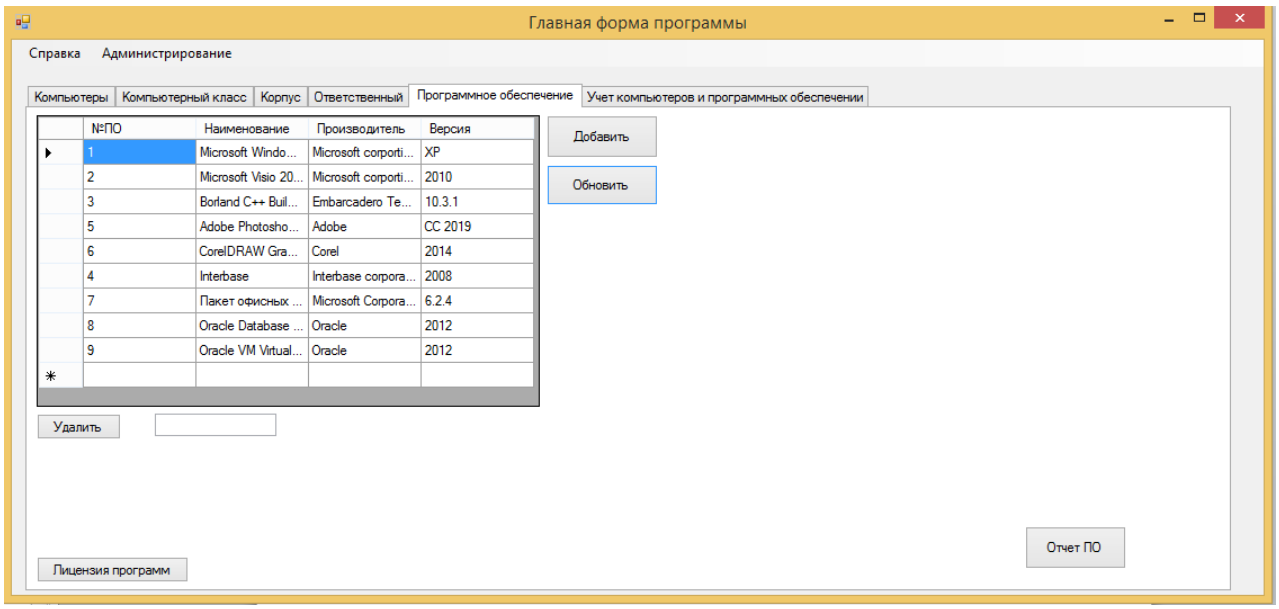


Рисунок 3.6 – Таблица «Программное обеспечение»

После просмотра программного обеспечения можно нажат на кнопку лицензия программ, посмотреть какие программы имею платный лицензии, а какие бесплатные (рисунок 3.7).

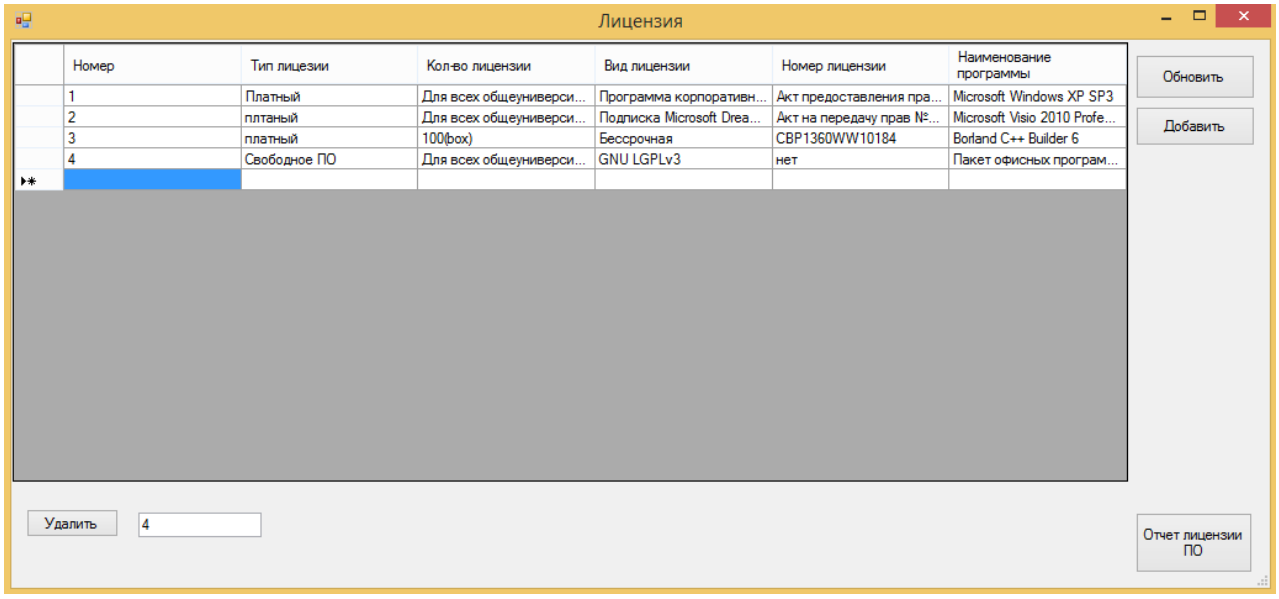


Рисунок 3.7 – Лицензия программ

Далее если пользователь захочет сделать отчет, по лицензиям программного обеспечения то может сделать это, нажав на кнопку «Отчет лицензии ПО» и при помощи табличного редактора MS Excel можно увидит отчет (рисунок 3.8).

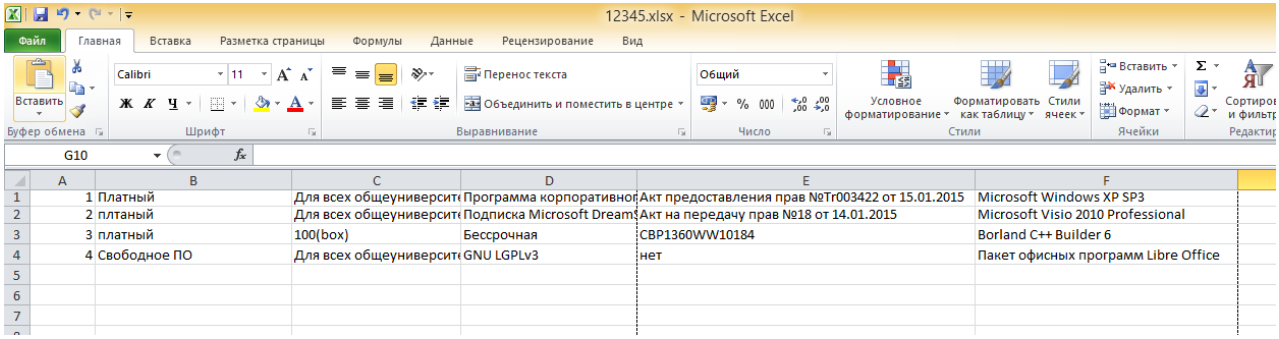


Рисунок 3.8 – Отчет о лицензии программного обеспечения

Далее переходим к главной вкладке приложения «Учет компьютерной техники и программного обеспечения» в этой вкладке находятся данные о компьютерах и о наименовании программного продукта в виде списка (рисукнок3.9).

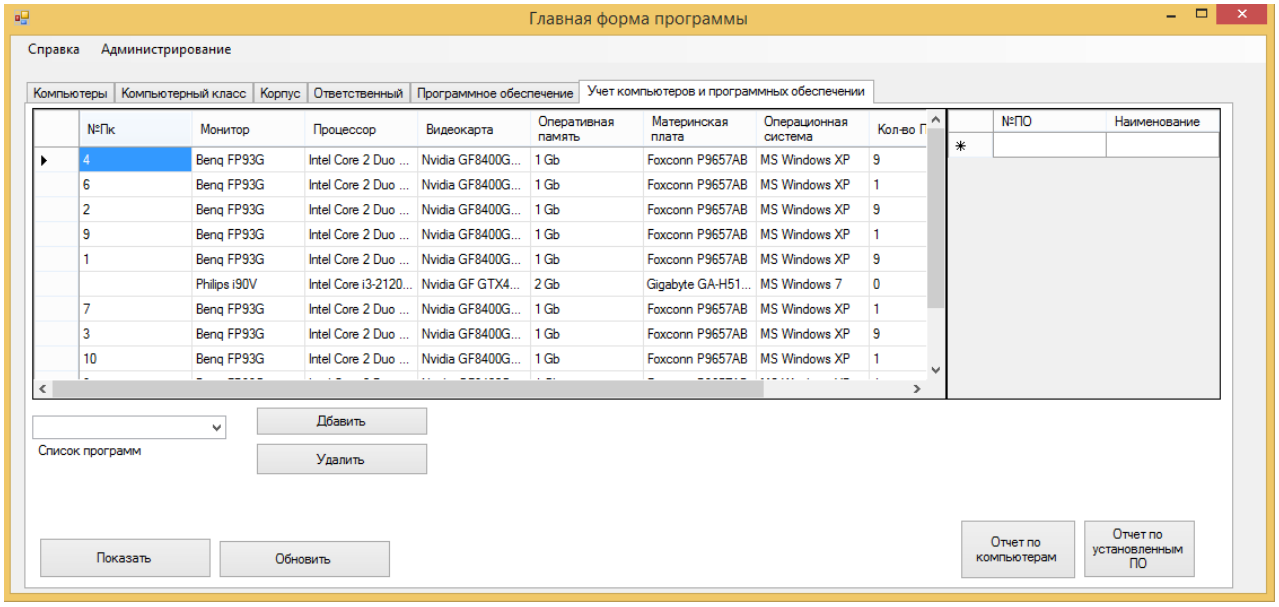


Рисунок 3.9 – Учет компьютеров и программного обеспечения

Поставив курсор на компьютер, и нажав на кнопку показать, пользователь может посмотреть какие программные продукты стоят на данном компьютере (рисунок 3.10).

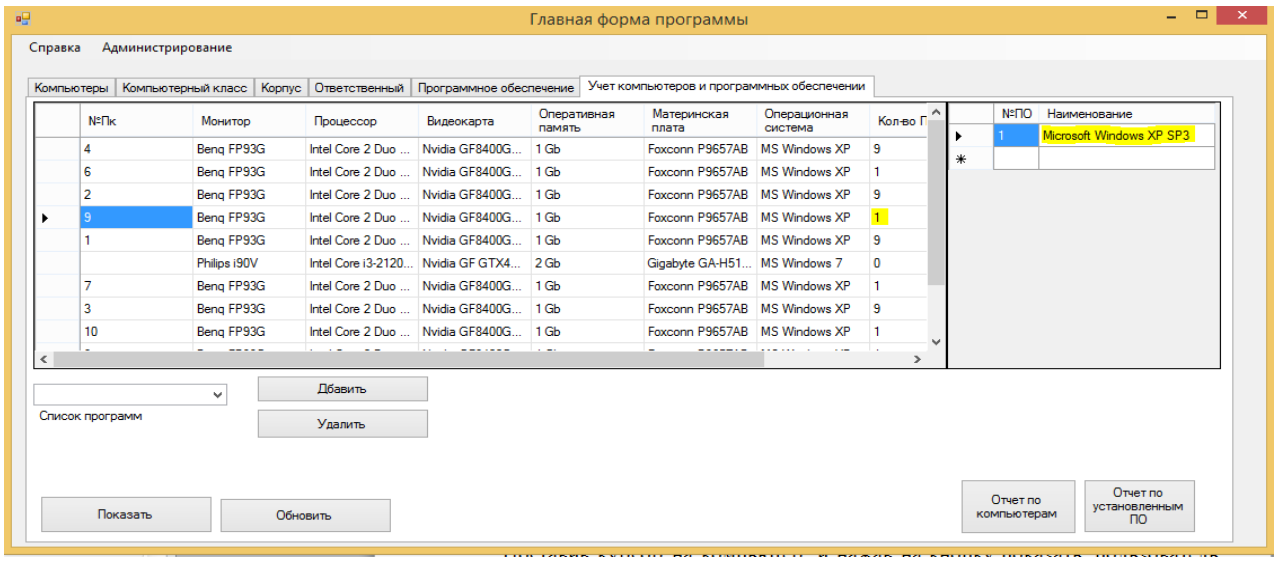


Рисунок 3.10 – Просмотр установленных ПО

Далее можно добавить или же удалить из списка программ, нужную пользователю программу для установки на компьютер или же удаления с компьютера на рисунках 3.11 и 3.12 это продемонстрирован.

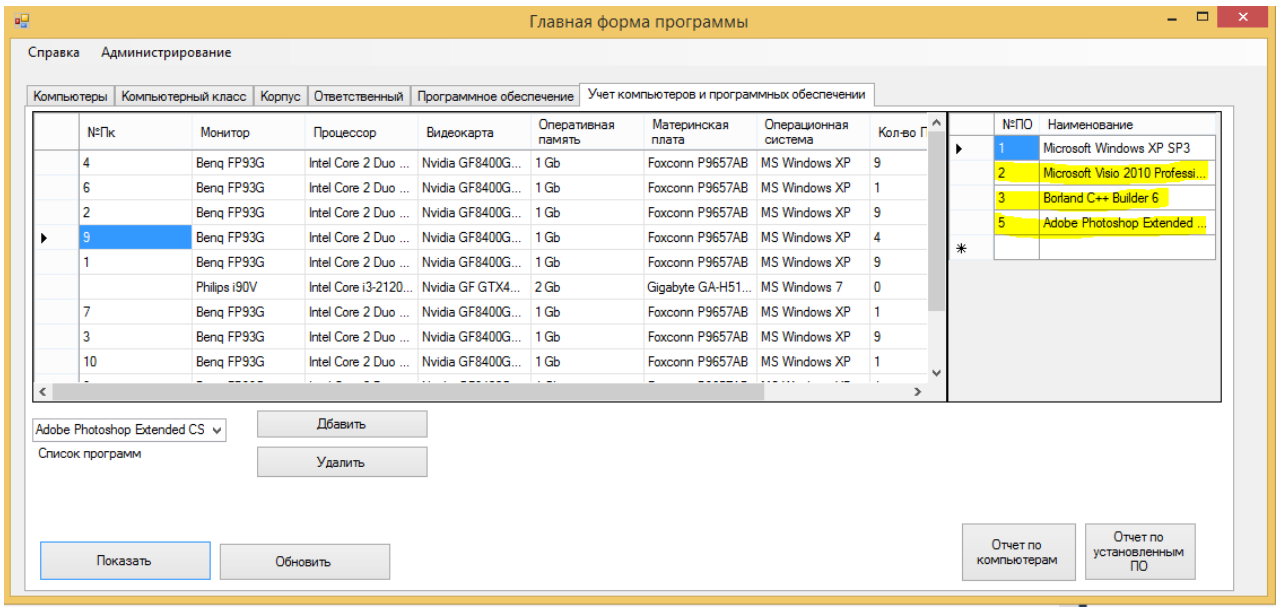


Рисунок 3.11 – Добавление программ на компьютер

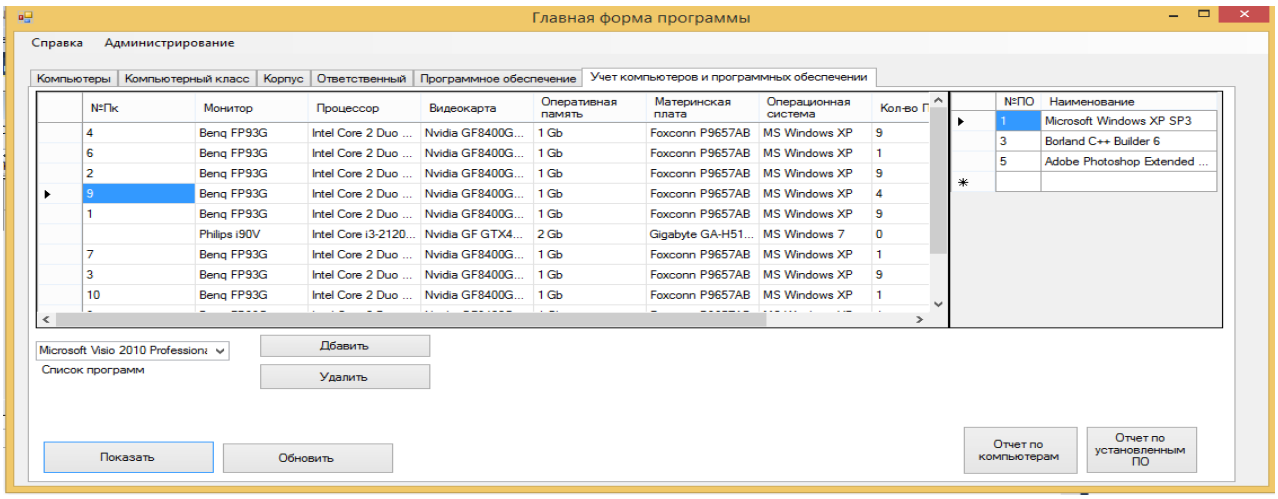
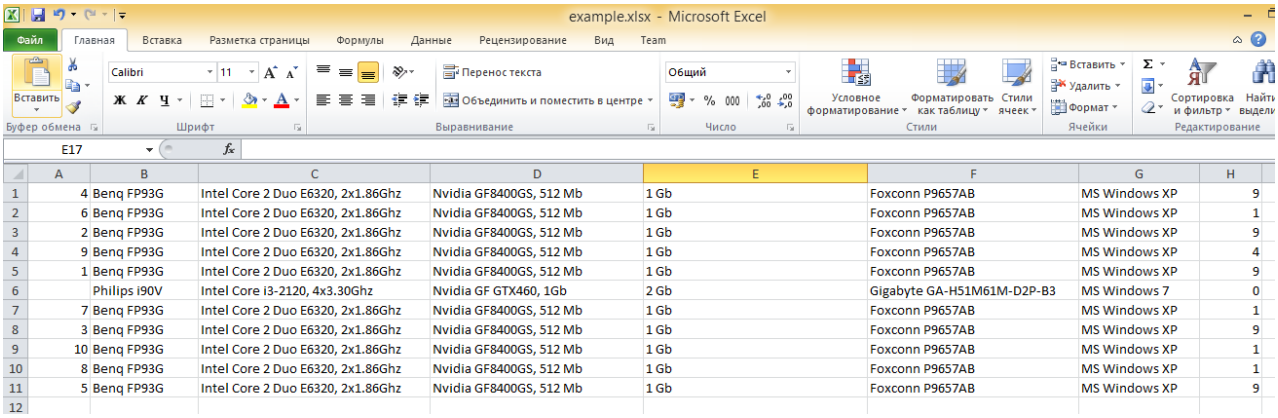


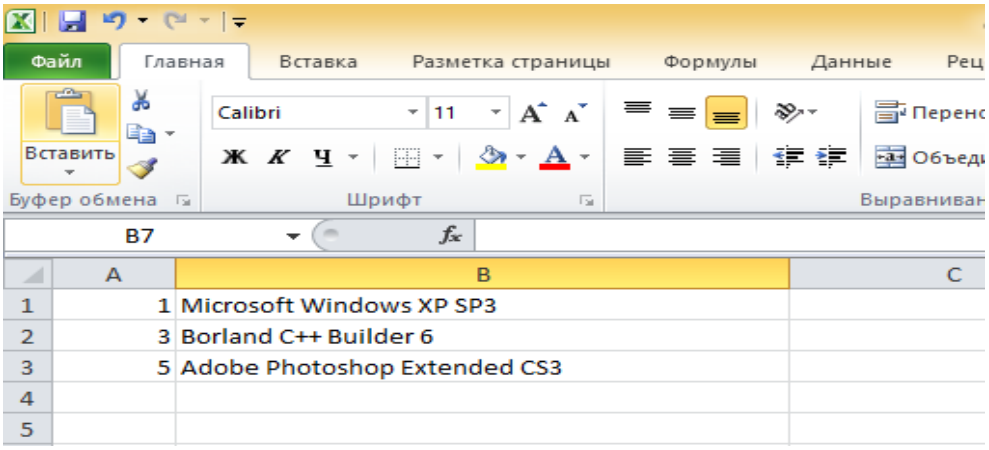
Рисунок 3.12 – Удаление программы с компьютера

После проделанной работы пользователь нажимая на кнопку «Отчет по компьютерам» и «Отчет по установленным ПО» на рисунках 3.13 и 3.14 это продемонстрировано.



	A	B	C	D	E	F	G	H
1	4	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	9
2	6	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	1
3	2	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	9
4	9	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	4
5	1	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	9
6		Philips i90V	Intel Core i3-2120, 4x3.30Ghz	Nvidia GF GTX460, 1Gb	2 Gb	Gigabyte GA-H51M61M-D2P-B3	MS Windows 7	0
7	7	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	1
8	3	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	9
9	10	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	1
10	8	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	1
11	5	Benq FP93G	Intel Core 2 Duo E6320, 2x1.86Ghz	Nvidia GF8400GS, 512 Mb	1 Gb	Foxconn P9657AB	MS Windows XP	9
12								

Рисунок 3.13 – Отчет по компьютерам



	A	B	C
1	1	Microsoft Windows XP SP3	
2	3	Borland C++ Builder 6	
3	5	Adobe Photoshop Extended CS3	
4			
5			

Рисунок 3.14 – Отчет по установленным ПО

Проведя, описание системы работоспособности ошибок выявлено не было. Можно сказать, что система работоспособна и оптимизирована.

Выводы по третьему разделу

Таким образом, можно подвести следующие выводы, что в данном разделе выпускной квалификационной работы была реализовано программная часть и описана информационная система.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была достигнута поставленная цель, а именно разработка информационной системы учета компьютерной техники и программного обеспечения в институте ИиЦТ НИУ «БелГУ» для совершенствования процесса учета.

В первом разделе выпускной квалификационной работы, был проведен анализ и описание предметной области и выбор средств разработки, после чего были выявленные требования к разрабатываемому проекту.

Во втором разделе выпускной квалификационной работы, было разработано техническое задание проекта и спроектированы модели.

В третьем разделе выпускной квалификационной работы была реализовано программная часть и описана информационная система.

Была проведена анализ, и описание предметной области который помог определить требования пользователей, выявить функциональные и нефункциональные требования к разрабатываемой информационной системе. Разработка технического задания определила основные функции, которая должна выполнять система. Программная реализация разработанных проектных моделей позволила получить информационную систему учета компьютерной техники и программного обеспечения, значительно упрощающую работу сотрудников института.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Сайт НИУ «БелГУ», Белгородский государственный национальный исследовательский университет [Электронный ресурс]. Режим доступа: https://www.bsu.edu.ru/bsu/

2 Клейменов, С.А. Администрирование в информационных системах / С.А. Клейменов, В.П. Мельников, А.М. Петраков. - Москва: Мир, 2008. - 272 с.

3 Аткинсон, Л. SQLite. Библиотека профессионала; М.: Вильямс, 2008. - 624 с.

4 Яргер, Р.Дж.; Риз, Дж.; Кинг, Т. MySQL и mSQL: Базы данных для небольших предприятий и Интернета; СПб: Символ-Плюс, 2013. - 560 с.

5 Уорсли, Дж. PostgreSQL. Для профессионалов (+ CD) / Дж. Уорсли, Дж. Дрейк. - М.: СПб: Питер, 2002. - 496 с.

6 Александреску, А. Язык программирования D / А. Александреску. — СПб: Символ-плюс, 2014. — 544 с.

7 Биллиг, В.А. Основы программирования на C#: Учебное пособие / В.А. Биллиг. - М.: Бином, 2012. - 483 с.

8 Ашарина, И.В. Основы программирования на языках C и C++: Курс лекций для высших учебных заведений / И.В. Ашарина. - М.: ГЛТ, 2012. - 208 с.

9 Берд, Барри Java для чайников / Барри Берд. - М.: Диалектика / Вильямс, 2013. - 521 с.

10 Боггс, М. UML и Rational Rose / М. Боггс. - Москва: РГГУ, 2016. - 438 с.

11 Дубейковский, В. И. Практика функционального моделирования с ALLFusion Process Modeler 4.1. Где? Зачем? Как? / В.И. Дубейковский. - М.: Диалог-Мифи, 2004. - 464 с.

12 Дубейковский, В. И. Эффективное моделирование с AllFusion Process Modeler / В.И. Дубейковский. - М.: Диалог-Мифи, 2007. - 384 с.

13 Грекул, В. И. Проектирование информационных систем / В. Грекул, Г.Н. Денищенко, Н.Л. Коровкина. – 2016. – 516 с.

14 Коваленко, В.В. Проектирование информационных систем / В.В. Коваленко. – СПБ: Форум, – 2015. – 320 с.

15 Губина Е. А. Проектирование информационной системы на основе связывания CASE-инструментария и реляционной базы данных / Е.А. Губина, Г.Х. Ирзаев Г, М.Г. Адеева. – 2014. – 71 с.

16 Муромцев В.В. Проектирование информационных систем / В.В. Муромцев. – Белгород.: БелГУ, – 2007. – 160 с.

17 Ипатова, Э. Р. Методологии и технологии системного проектирования информационных систем / Э.Р. Ипатова, Ю.В. Ипатов. - М.: Флинта, 2016. - 256 с.

18 Исаев, Г.Н. Проектирование информационных систем: Учебное пособие / Г.Н. Исаев. - М.: Омега-Л, 2013. - 424 с.

19 Михелёв В.М. Базы данных и СУБД / В.М. Михелёв. – Белгород.: БелГУ, – 2007. – 200 с.

20 Атре, Ш. Структурный подход к организации баз данных / Ш. Атре. - М.: Финансы и статистика, 2010. - 317 с.

21 Редько, В.Н. Базы данных и информационные системы / В.Н. Редько, И.А. Бассараб. - М.: Знание, 2004. - 240 с.

22 Саймон, Ригс Администрирование PostgreSQL 9. Книга рецептов / Ригс Саймон. - М.: ДМК Пресс, 2018. - 806 с.

23 Стоунз PostgreSQL. Основы / Стоунз, Мэттью Ричард; , Нейл. - М.: СПб: Символ-Плюс, 2002. - 640 с.

24 Биллиг, В. Основы программирования на C# / В. Биллиг. - М.: Бином. Лаборатория знаний, 2006. - 483 с.

ПРИЛОЖЕНИЯ А

Исходный код программы

```
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 Npgsql;
using Excel = Microsoft.Office.Interop.Excel;

namespace WindowsFormsApplication2
{
   public partial class Form6 : Form
   {
      string idn;
      public Form6()
      {
         InitializeComponent();
         dataGridView1.Rows.Clear();
         string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
         NpgsqlConnection connect = new NpgsqlConnection(conn);
         connect.Open();
         NpgsqlCommand command = new NpgsqlCommand("SELECT computer.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram,computer.materinskaya_plata, computer.os, auditoriya.nomer_aud FROM computer,auditoriya
WHERE  computer.id_aud = auditoriya.id_aud ", connect);
         NpgsqlDataReader reader;
         reader = command.ExecuteReader();
         while (reader.Read())
         {
            string[] s = new string[10];
            for (int i = 0; i < reader.FieldCount; i++)
            {
               s[i] = reader[i].ToString();
            }
            dataGridView1.Rows.Add(s);
         }
         dataGridView2.Rows.Clear();
         NpgsqlCommand command1 = new NpgsqlCommand("SELECT auditoriya.id_aud, auditoriya.nomer_aud,
korpus.nomer_kor, otvetstvenniy.fio FROM auditoriya,korpus,otvetstvenniy WHERE  auditoriya.id_kor = korpus.id_kor and
auditoriya.id_otvets = otvetstvenniy.id_ot order by auditoriya.id_aud ASC", connect);
         NpgsqlDataReader reader1;
         reader1 = command1.ExecuteReader();
         while (reader1.Read())
         {
            string[] s = new string[6];
            for (int i = 0; i < reader1.FieldCount; i++)
            {
               s[i] = reader1[i].ToString();
            }
            dataGridView2.Rows.Add(s);
         }
         dataGridView3.Rows.Clear();
         NpgsqlCommand command3 = new NpgsqlCommand("select korpus.nomer_kor,korpus.adress from korpus", connect);
         NpgsqlDataReader reader2;
         reader2 = command3.ExecuteReader();
         while (reader2.Read())
         {
            string[] s = new string[8];
            for (int i = 0; i < reader2.FieldCount; i++)
```

```
            {
                s[i] = reader2[i].ToString();
            }
            dataGridView3.Rows.Add(s);
        }
        dataGridView4.Rows.Clear();
        NpgsqlCommand command4 = new NpgsqlCommand("select
otvetstvenniy.fio,otvetstvenniy.god_rojdeniya,otvetstvenniy.nomer_telephona, otvetstvenniy.dolzhnost from otvetstvenniy ",
connect);
        NpgsqlDataReader reader3;
        reader3 = command4.ExecuteReader();
        while (reader3.Read())
        {
            string[] s = new string[7];
            for (int i = 0; i < reader3.FieldCount; i++)
            {
                s[i] = reader3[i].ToString();
            }
            dataGridView4.Rows.Add(s);
        }
        dataGridView7.Rows.Clear();
        NpgsqlCommand comande5 = new NpgsqlCommand("select * from po", connect);
        NpgsqlDataReader reader4;
        reader4 = command4.ExecuteReader();
        while (reader4.Read())
        {
            string[] s = new string[7];
            for (int i = 0; i < reader4.FieldCount; i++)
            {
                s[i] = reader4[i].ToString();
            }
            dataGridView7.Rows.Add(s);
        }
        dataGridView5.Rows.Clear();
        NpgsqlCommand comande6 = new NpgsqlCommand("SELECT uchet.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram,computer.materinskaya_plata,computer.os, COUNT(uchet.id_po) AS Count_prog FROM
computer LEFT JOIN uchet  ON computer.id_pk = uchet.id_pk   GROUP BY uchet.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram,computer.materinskaya_plata,computer.os;", connect);
        NpgsqlDataReader reader5;
        reader5 = comande6.ExecuteReader();
        while (reader5.Read())
        {
            // DataTable data = new DataTable();
            // IDbDataAdapter adapter = new IDbDataAdapter(comande6);
            // adapter.Fill()
            string[] s = new string[10];
            for (int i = 0; i < reader5.FieldCount; i++)
            {
                s[i] = reader5[i].ToString();
            }
            dataGridView5.Rows.Add(s);
        }


        NpgsqlCommand comanD = new NpgsqlCommand("SELECT name_po FROM PO", connect);
        NpgsqlDataReader readeR;
        readeR = comanD.ExecuteReader();
        while (readeR.Read())
        {
            comboBox1.Items.Add(readeR["name_po"].ToString());
        }
        connect.Close();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        int row = dataGridView5.SelectedCells[0].RowIndex;
        string key = dataGridView5[0, row].FormattedValue.ToString();
        if (key != "")
        {
```

```
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            dataGridView6.Rows.Clear();
            NpgsqlCommand comande6 = new NpgsqlCommand("select uchet.id_po, po.name_po from uchet, po WHERE
uchet.id_po = po.id_po AND uchet.id_pk = " + key + ";", connect);
            NpgsqlDataReader reader5;
            reader5 = comande6.ExecuteReader();
            while (reader5.Read())
            {
               string[] s = new string[7];
               for (int i = 0; i < reader5.FieldCount; i++)
               {
                  s[i] = reader5[i].ToString();
               }
               dataGridView6.Rows.Add(s);
            }
            connect.Close();
         }
         else
         {
            DialogResult result = MessageBox.Show("Не найден идентификатор", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
         }
      }

      private void button2_Click(object sender, EventArgs e)
      {
         if (comboBox1.Text != "")
         {
            int row = dataGridView5.SelectedCells[0].RowIndex;
            string key = dataGridView5[0, row].FormattedValue.ToString();
            if (key != "")
            {
               string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
               NpgsqlConnection connect = new NpgsqlConnection(conn);
               connect.Open();
               NpgsqlCommand comande6 = new NpgsqlCommand("SELECT id_po FROM PO WHERE name_po = '" +
comboBox1.Text + "'", connect);
               NpgsqlDataReader reader5;


               reader5 = comande6.ExecuteReader();
               while (reader5.Read())
               {
                  idn = reader5["id_po"].ToString();
               }
               NpgsqlCommand C = new NpgsqlCommand("INSERT INTO uchet(id_pk, id_po) VALUES(" + key + "," + idn + ");",
connect);
               NpgsqlDataReader R;
               R = C.ExecuteReader();
               while (R.Read())
               {

               }
               connect.Close();
            }
            else
            {
               DialogResult result = MessageBox.Show("Не найден идентификатор", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
            }

         }
         else
         {
            DialogResult result = MessageBox.Show("Выбирите программу", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
         }
      }
```

```
        private void button3_Click(object sender, EventArgs e)
        {
            if (comboBox1.Text != "")
            {
                int row = dataGridView5.SelectedCells[0].RowIndex;
                string key = dataGridView5[0, row].FormattedValue.ToString();
                if (key != "")
                {
                    string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
                    NpgsqlConnection connect = new NpgsqlConnection(conn);
                    connect.Open();
                    NpgsqlCommand comande6 = new NpgsqlCommand("SELECT id_po FROM PO WHERE name_po = '" +
comboBox1.Text + "'", connect);
                    NpgsqlDataReader reader5;


                    reader5 = comande6.ExecuteReader();
                    while (reader5.Read())
                    {
                        idn = reader5["id_po"].ToString();
                    }
                    NpgsqlCommand C = new NpgsqlCommand("DELETE FROM uchet WHERE id_pk = " + key + " AND id_po = " +
idn + ";", connect);
                    NpgsqlDataReader R;
                    R = C.ExecuteReader();
                    while (R.Read())
                    {

                    }
                    connect.Close();
                }
                else
                {
                    DialogResult result = MessageBox.Show("Не найден идентификатор", "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
                }
            }
        }


        private void button4_Click_1(object sender, EventArgs e)
        {
            Form9 form = new Form9();
            form.ShowDialog();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            dataGridView7.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command = new NpgsqlCommand("select * from po", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
                string[] s = new string[7];
                for (int i = 0; i < reader.FieldCount; i++)
                {
                    s[i] = reader[i].ToString();
                }
                dataGridView7.Rows.Add(s);
            }
            connect.Close();
        }

        private void button6_Click(object sender, EventArgs e)
        {
```

```
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command3 = new NpgsqlCommand("DELETE FROM PO WHERE id_po  =" + textBox5.Text + "",
connect);
            command3.ExecuteNonQuery();
            connect.Close();
        }

        private void button7_Click(object sender, EventArgs e)
        {
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command3 = new NpgsqlCommand("DELETE FROM otvetstvenniy WHERE fio ='" + textBox6.Text +
"'", connect);
            command3.ExecuteNonQuery();
            connect.Close();
        }

        private void button8_Click(object sender, EventArgs e)
        {
            Form8 form = new Form8();
            form.ShowDialog();
        }

        private void button9_Click(object sender, EventArgs e)
        {
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            dataGridView4.Rows.Clear();
            NpgsqlCommand command4 = new NpgsqlCommand("select * from otvetstvenniy", connect);
            NpgsqlDataReader reader3;
            reader3 = command4.ExecuteReader();
            while (reader3.Read())
            {
                string[] s = new string[7];
                for (int i = 0; i < reader3.FieldCount; i++)
                {
                    s[i] = reader3[i].ToString();
                }
                dataGridView4.Rows.Add(s);
            }
        }

        private void button10_Click(object sender, EventArgs e)
        {
            Form7 form = new Form7();
            form.ShowDialog();
        }

        private void button11_Click(object sender, EventArgs e)
        {
            Form5 form = new Form5();
            form.ShowDialog();
        }

        private void button12_Click(object sender, EventArgs e)
        {
            Form3 form = new Form3();
            form.ShowDialog();
        }

        private void button13_Click(object sender, EventArgs e)
        {
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
```

```
            NpgsqlCommand command3 = new NpgsqlCommand("DELETE FROM computer WHERE id_pk =" + textBox23.Text +
"", connect);
            command3.ExecuteNonQuery();
            connect.Close();
        }

        private void button14_Click(object sender, EventArgs e)
        {
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            dataGridView5.Rows.Clear();
            NpgsqlCommand comande6 = new NpgsqlCommand("SELECT uchet.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram, computer.materinskaya_plata,computer.os, COUNT(uchet.id_po) AS Count_prog FROM
computer LEFT JOIN uchet  ON computer.id_pk = uchet.id_pk GROUP BY uchet.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram, computer.materinskaya_plata,computer.os;", connect);
            NpgsqlDataReader reader5;
            reader5 = comande6.ExecuteReader();
            while (reader5.Read())
            {
                // DataTable data = new DataTable();
                // IDbDataAdapter adapter = new IDbDataAdapter(comande6);
                // adapter.Fill()
                string[] s = new string[9];
                for (int i = 0; i < reader5.FieldCount; i++)
                {
                    s[i] = reader5[i].ToString();
                }
                dataGridView5.Rows.Add(s);
            }
            connect.Close();
        }

        private void button15_Click(object sender, EventArgs e)
        {
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command3 = new NpgsqlCommand("DELETE FROM korpus WHERE id_kor =" + textBox25.Text + "",
connect);
            command3.ExecuteNonQuery();
            connect.Close();
        }

        private void button16_Click(object sender, EventArgs e)
        {
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command3 = new NpgsqlCommand("DELETE FROM auditoriya WHERE id_aud =" + textBox26.Text +
"", connect);
            command3.ExecuteNonQuery();
            connect.Close();
        }



        private void oПрограммеToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Справка1 form = new Справка1();
            form.ShowDialog();
        }

        private void button21_Click(object sender, EventArgs e)
        {
            dataGridView1.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
```

```
            NpgsqlCommand command = new NpgsqlCommand("SELECT computer.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram,computer.materinskaya_plata, computer.os, auditoriya.nomer_aud FROM computer,auditoriya
WHERE  computer.id_aud = auditoriya.id_aud ", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
               string[] s = new string[9];
               for (int i = 0; i < reader.FieldCount; i++)
               {
                  s[i] = reader[i].ToString();
               }
               dataGridView1.Rows.Add(s);
            }
        }

        private void button22_Click(object sender, EventArgs e)
        {
            dataGridView2.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command = new NpgsqlCommand("SELECT auditoriya.id_aud, auditoriya.nomer_aud,
korpus.nomer_kor, otvetstvenniy.fio FROM auditoriya,korpus,otvetstvenniy WHERE  auditoriya.id_kor = korpus.id_kor and
auditoriya.id_otvets = otvetstvenniy.id_ot order by auditoriya.id_aud ASC", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
               string[] s = new string[7];
               for (int i = 0; i < reader.FieldCount; i++)
               {
                  s[i] = reader[i].ToString();
               }
               dataGridView2.Rows.Add(s);
            }
        }

        private void button23_Click(object sender, EventArgs e)
        {
            dataGridView3.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command = new NpgsqlCommand("select korpus.nomer_kor, korpus.adress from korpus", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
               string[] s = new string[7];
               for (int i = 0; i < reader.FieldCount; i++)
               {
                  s[i] = reader[i].ToString();
               }
               dataGridView3.Rows.Add(s);
            }
        }

        private void button24_Click(object sender, EventArgs e)
        {
            dataGridView4.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command = new NpgsqlCommand("select otvetstvenniy.fio, otvetstvenniy.god_rojdeniya,
otvetstvenniy.nomer_telephona,otvetstvenniy.dolzhnost from otvetstvenniy", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
```

```
            string[] s = new string[7];
            for (int i = 0; i < reader.FieldCount; i++)
            {
                s[i] = reader[i].ToString();
            }
            dataGridView4.Rows.Add(s);
        }
    }

    private void лицензияToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Лицензия form = new Лицензия();
        form.ShowDialog();
    }

    private void button9_Click_1(object sender, EventArgs e)
    {
        dataGridView4.Rows.Clear();
        string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
        NpgsqlConnection connect = new NpgsqlConnection(conn);
        connect.Open();
        NpgsqlCommand command = new NpgsqlCommand("select otvetstvenniy.fio, otvetstvenniy.god_rojdeniya,
otvetstvenniy.nomer_telephona,otvetstvenniy.dolzhnost from otvetstvenniy where (fio LIKE '%" + textBox1.Text + "%' )",
connect);
        NpgsqlDataReader reader;
        reader = command.ExecuteReader();
        while (reader.Read())
        {
            string[] s = new string[12];
            for (int i = 0; i < reader.FieldCount; i++)
            {
                s[i] = reader[i].ToString();
            }
            dataGridView4.Rows.Add(s);
        }

    }

    private void button25_Click(object sender, EventArgs e)
    {
        dataGridView3.Rows.Clear();
        string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
        NpgsqlConnection connect = new NpgsqlConnection(conn);
        connect.Open();
        NpgsqlCommand command = new NpgsqlCommand("SELECT korpus.nomer_kor, korpus.adress from korpus WHERE
nomer_kor =" + textBox2.Text + "", connect);
        NpgsqlDataReader reader;
        reader = command.ExecuteReader();
        while (reader.Read())
        {
            string[] s = new string[12];
            for (int i = 0; i < reader.FieldCount; i++)
            {
                s[i] = reader[i].ToString();
            }
            dataGridView3.Rows.Add(s);
        }
    }

    private void button26_Click(object sender, EventArgs e)
    {
        dataGridView2.Rows.Clear();
        string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
        NpgsqlConnection connect = new NpgsqlConnection(conn);
        connect.Open();
        NpgsqlCommand command = new NpgsqlCommand("SELECT auditoriya.id_aud ,auditoriya.nomer_aud,
korpus.nomer_kor, otvetstvenniy.fio from auditoriya ,korpus, otvetstvenniy WHERE   (auditoriya.nomer_aud LIKE '%" +
textBox3.Text + "%')", connect);
        NpgsqlDataReader reader;
        reader = command.ExecuteReader();
```

```
            while (reader.Read())
            {
                string[] s = new string[12];
                for (int i = 0; i < reader.FieldCount; i++)
                {
                    s[i] = reader[i].ToString();
                }
                dataGridView2.Rows.Add(s);
            }
        }

        private void button27_Click(object sender, EventArgs e)
        {
            dataGridView1.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command = new NpgsqlCommand("SELECT computer.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram,computer.materinskaya_plata, computer.os, auditoriya.nomer_aud FROM computer,auditoriya
WHERE  computer.id_aud = auditoriya.id_aud   AND     auditoriya.nomer_aud ='" + textBox4.Text + "'", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
                string[] s = new string[12];
                for (int i = 0; i < reader.FieldCount; i++)
                {
                    s[i] = reader[i].ToString();
                }
                dataGridView1.Rows.Add(s);
            }
        }

        private void button28_Click(object sender, EventArgs e)
        {
            Лицензия form = new Лицензия();
            form.ShowDialog();
        }

        private void выходToolStripMenuItem_Click(object sender, EventArgs e)
        {

        }

        private void сотрудникиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form1 form = new Form1();
            form.ShowDialog();
        }

        private void printDocument1_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)
        {
            Bitmap bm = new Bitmap(this.dataGridView1.Width, this.dataGridView1.Height);
            dataGridView1.DrawToBitmap(bm, new Rectangle(0, 0, this.dataGridView1.Width, this.dataGridView1.Height));
            e.Graphics.DrawImage(bm, 0, 0);
        }

        private void button20_Click(object sender, EventArgs e)
        {
            printDocument1.Print();
        }

        private void button17_Click(object sender, EventArgs e)
        {
            Microsoft.Office.Interop.Excel.Application XlApp = new Microsoft.Office.Interop.Excel.Application();
            Microsoft.Office.Interop.Excel.Workbook XlWorkBook =
XlApp.Workbooks.Open(@"C:\Users\Majestick\Documents\example.xlsx"); //создать новый файл: XlApp.Workbooks.Add();
            Microsoft.Office.Interop.Excel.Worksheet XlWorkSheet =
(Microsoft.Office.Interop.Excel.Worksheet)XlWorkBook.Worksheets.get_Item(1); //1-й лист по порядку
            for (int i = 0; i < dataGridView1.Rows.Count; i++)
```

```
            {
                for (int j = 0; j < dataGridView1.ColumnCount; j++)
                {
                    XlWorkSheet.Cells[i + 1, j + 1] = dataGridView1.Rows[i].Cells[j].Value;
                }
            }
            XlApp.Visible = true;
        }

        private void button29_Click(object sender, EventArgs e)
        {
            Microsoft.Office.Interop.Excel.Application XlApp = new Microsoft.Office.Interop.Excel.Application();
            Microsoft.Office.Interop.Excel.Workbook XlWorkBook =
XlApp.Workbooks.Open(@"C:\Users\Majestick\Documents\example.xlsx"); //создать новый файл: XlApp.Workbooks.Add();
            Microsoft.Office.Interop.Excel.Worksheet XlWorkSheet =
(Microsoft.Office.Interop.Excel.Worksheet)XlWorkBook.Worksheets.get_Item(1); //1-й лист по порядку
            for (int i = 0; i < dataGridView1.Rows.Count; i++)
            {
                for (int j = 0; j < dataGridView5.ColumnCount; j++)
                {
                    XlWorkSheet.Cells[i + 1, j + 1] = dataGridView5.Rows[i].Cells[j].Value;
                }
            }
            XlApp.Visible = true;
        }

        private void button30_Click(object sender, EventArgs e)
        {
            Microsoft.Office.Interop.Excel.Application XlApp = new Microsoft.Office.Interop.Excel.Application();
            Microsoft.Office.Interop.Excel.Workbook XlWorkBook =
XlApp.Workbooks.Open(@"C:\Users\Majestick\Documents\example.xlsx"); //создать новый файл: XlApp.Workbooks.Add();
            Microsoft.Office.Interop.Excel.Worksheet XlWorkSheet =
(Microsoft.Office.Interop.Excel.Worksheet)XlWorkBook.Worksheets.get_Item(1); //1-й лист по порядку
            for (int i = 0; i < dataGridView6.Rows.Count; i++)
            {
                for (int j = 0; j < dataGridView6.ColumnCount; j++)
                {
                    XlWorkSheet.Cells[i + 1, j + 1] = dataGridView6.Rows[i].Cells[j].Value;
                }
            }
            XlApp.Visible = true;
        }

        private void button31_Click(object sender, EventArgs e)
        {
            dataGridView5.Rows.Clear();
            string conn = "Server=127.0.0.1;Port=5432;User Id=postgres;Password=934494699!;Database=postgres";
            NpgsqlConnection connect = new NpgsqlConnection(conn);
            connect.Open();
            NpgsqlCommand command = new NpgsqlCommand("SELECT computer.id_pk, computer.monitor, computer.proc,
computer.videokarta, computer.ram,computer.materinskaya_plata, computer.os, auditoriya.nomer_aud FROM computer,auditoriya
WHERE  computer.id_aud = auditoriya.id_aud   AND     auditoriya.nomer_aud ='" + textBox4.Text + "'", connect);
            NpgsqlDataReader reader;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
                string[] s = new string[12];
                for (int i = 0; i < reader.FieldCount; i++)
                {
                    s[i] = reader[i].ToString();
                }
                dataGridView5.Rows.Add(s);
            }


        }

        private void button20_Click_1(object sender, EventArgs e)
        {
            Microsoft.Office.Interop.Excel.Application XlApp = new Microsoft.Office.Interop.Excel.Application();
```

```
        Microsoft.Office.Interop.Excel.Workbook XlWorkBook =
XlApp.Workbooks.Open(@"C:\Users\Majestick\Documents\example.xlsx"); //создать новый файл: XlApp.Workbooks.Add();
        Microsoft.Office.Interop.Excel.Worksheet XlWorkSheet =
(Microsoft.Office.Interop.Excel.Worksheet)XlWorkBook.Worksheets.get_Item(1); //1-й лист по порядку
        for (int i = 0; i < dataGridView7.Rows.Count; i++)
        {
           for (int j = 0; j < dataGridView7.ColumnCount; j++)
           {
              XlWorkSheet.Cells[i + 1, j + 1] = dataGridView7.Rows[i].Cells[j].Value;
           }
        }
        XlApp.Visible = true;
     }
  }
}
```