МИНИСТЕРСТВО ОБРАЗОВАНИЯ ОРЕНБУРГСКОЙ ОБЛАСТИ **ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

«**ОРЕНБУРГСКИЙ КОЛЛЕДЖ ЭКОНОМИКИ И ИНФОРМАТИКИ**»

**(ГАПОУ СПО ОКЭИ)**

**ОТЧЁТ**

ОКЭИ 09.02.07. 7019. 15 П

ПП.06.01 Производственная практика

По модулю

Количество листов: 76

Дата готовности: 27.09.2020

Разработала: студентка группы 4ис1 Никифорова А.С.

Руководитель: Асташева Татьяна Сергеевна.

Соответствие отчета с заданием на практику \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Защищён \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оренбург 2020

**Содержание**

|  |  |
| --- | --- |
| Введение………………………………………………………………………….. | 3 |
| Характеристика предприятия………………………………………................. | 8 |
| Техническое задание……………………………………………………........... | 9 |
| Информационное обеспечение……………………………………….............. | 17 |
| Описание автоматизированных функций……………………………............ | 27 |
| Руководство программиста………………….……………………………....... | 28 |
| Руководство пользователя…………………………………………………….. | 31 |
| Тестирование и отладка программного комплекса……………….…………. | 35 |
| Техника безопасности и пожарная безопасность……………………………. | 37 |
| Заключение………………………………………………………………………..  Список использованных источников…………………………………………....  Приложение А (обязательное) Расчет экономической эффективности……….  Приложение Б (обязательное) Входные документы……………………………  Приложение В (обязательное) Выходные документы………………………….  Приложение Г (обязательное) Информационная модель………………………  Приложение Д (обязательное) Функциональная модель………………………  Приложение Е (обязательное) Журнал опытной эксплуатации……………….  Приложение Ж (обязательное) Диаграмма прецедентов………………………  Приложение Л (обязательное) Диаграмма деятельности ……………………  Приложение И (обязательное) Листинг программы …………………………. | 40 |
| 43 |
| 45 |
| 51 |
| 54 |
| 56 |
| 57 |
| 60 |
| 61  62 |
| 65 |
| Приложение К (обязательное) Контрольный пример ……………………… | 74 |

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Введение**

В нашем мире всё больше растет потребность в совершенствовании технологий и программного обеспечения к ним, а также основой этого совершенствования является потребность общаться с людьми из далеких уголков земли, рассчитывать траекторию полета самолета, исследовать космос так же не возможно без цифровых технологий и т.д. Мир цифровых технологий позволит не только услышать собеседника, но и увидеть. Вы можете сидя дома посмотреть, изучить различного рода информацию при помощи Интернета.

Потоки информации, циркулирующие в мире, который нас окружает, огромны. Во времени они имеют тенденцию к увеличению. Поэтому в любой организации, как большой, так и маленькой, возникает проблема такой организации управления данными, которая обеспечила бы наиболее эффек­тивную работу. Некоторые организации используют для этого шкафы с папками, но большинство предпочитают компьютеризированные способы – базы данных, позволяющие эффективно хранить, структурировать и систематизировать большие объемы дан­ных. И уже сегодня без баз данных невозможно представить работу большинства финансовых, промышленных, торговых и прочих организаций. Не будь баз данных, они бы просто захлебнулись в информационной лавине.

Существует много веских причин перевода существующей информации на компьютерную основу. Сейчас стоимость хранения информации в файлах ЭВМ дешевле, чем на бумаге. Базы данных позволяют хранить, структурировать информацию и извлекать оптимальным для пользователя образом. Использование клиент/серверных технологий позволяют сберечь значительные средства, а главное и время для получения необходимой информации, а также упрощают доступ и ведение, поскольку они основываются на комплексной обработке данных и централизации их хранения. Кроме того, ЭВМ позволяет хранить любые форматы данных текст, чертежи, данные в рукописной форме, фотографии, записи голоса и т.д.

Для использования столь огромных объемов хранимой информации, помимо развития системных устройств, средств передачи данных, памяти необходимы средства обеспечения диалога человек-ЭВМ, которые позволяют пользователю вводить запросы, читать файлы, модифицировать хранимые данные, добавлять новые данные или принимать решения на основании хранимых данных. Для обеспечения этих функций созданы специализированные средства – системы управления базами данных (СУБД). Современные СУБД - многопользовательские системы управления базой данных, которые специализируется на управлении массивом ин­формации одним или множеством одновременно работающих пользователей.

Исходя из этого, проблему, которую я хочу поднять, нельзя назвать не актуальной. Каждое предприятие, имеющее компьютеры, старается более или менее автоматизировать работу своих специалистов, тем самым, упрощая и ускоряя их работу.

Современную конъюнктуру абсолютного большинства организаций медицинской сферы и сферы социального обслуживания невозможно представить без средств автоматизации. Различного рода системы на базе компьютерной техники, телекоммуникационного оборудования и информационных технологий способствуют повышению производительности труда, повышению качества и надёжности оказания медицинских услуг и коренному повышению качества оказания медицинских услуг.

Для успешной деятельности в любой сфере необходима информация, при чём качество этой информации и скорость ее получения в некоторых случаях может быть критичной. В медицинской сфере своевременность получения информации является «жизненно» необходимой. Возрастают объемы этой информации, ее сложность, и интеллектуальные возможности человека не успевают следовать этой тенденции, а возможности ее ручной переработки остаются прежними. Выходом из сложившейся ситуации может стать использование автоматизированных информационных систем (АИС), берущих на себя огромную часть работ, связанных с учетом, хранением и обработкой информации, а также дающих возможность человеку адекватно оценить ситуацию и принять верное решение. Неотъемлемой частью таких АИС является база данных (БД), хранящая всю информацию, необходимую для учета аспектов различного рода деятельности, и приложение, решающее определённые задачи. Таким образом, создание АИС «Учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум» является необходимым для нормального функционирования медицинской организации.

Актуальность задачи. С ростом числа используемой компьютерной техники в организации возникла проблема учёта, как компьютеров, так и отдельных комплектующих, а также проводимых технических осмотров и ремонтов. Хранить информацию и производить её выборку без специально разработанной автоматизированной информационной системы очень сложно. Технические осмотры и ремонт производятся несвоевременно, запасные детали не закупаются вовремя. Руководство организации не получает реальной картины по состоянию ИТ инфраструктуры.

Хранение необходимой информации в виде базы данных на сервере, безусловно, более целесообразно, чем хранение их в виде файлов электронных таблиц или, тем более в виде бумаг. Значительно упрощается поиск нужной информации, имеется возможность хранить данные за большой временной отрезок, упрощается процесс составления отчётов. А главное, использование баз данных позволяет хранить структурированные, не противоречивые данные, минимально избыточные и целостные.

Целью данной работы является разработка автоматизированной информационной системы учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». С помощью которой упростится поиск процесс учета заявок на ремонт техники, поиск информации и работа с документами, как для специалистов, так и для других сотрудников.

Для достижения поставленной цели необходимо решить ряд инженерных задач по специальности:

* выполнить анализ деятельности организации;
* выявить информационные процессы подлежащие автоматизации и описать их;
* выполнить анализ IT-составляющей деятельности организации и определить перспективные направления развития производства процесс учета заявок на ремонт компьютерной техники в организации;
* провести анализ аналогов задачи автоматизации информационных процессов;
* разработать архитектуру АИС;
* произвести выбор средств программирования и СУБД для разработки АИС;
* провести анализ информационных потоков и разработать модели данных АИС;
* разработать алгоритмы приложения;
* протестировать разработанную АИС;
* рассмотреть вопросы охраны труда;
* рассчитать экономическую эффективность проекта.

Для реализации поставленных целей система должна решать следующие задачи:

1. Обеспечить авторизованный вход пользователей в систему;
2. Обеспечить регистрацию, изменение и просмотр информации о заявках на ремонт компьютерной техники;
3. Ведение учета компьютерного оборудования;
4. Ведение необходимых для работы системы справочников;
5. Редактирование данных в информационной системе;
6. Построение необходимых отчётов.

Объектом исследования данной работы является ГАПОУ «Акбулакский политехнический техникум».

Предметом исследования в работе являются – автоматизация производственных функций сотрудников отдела компьютерного обеспечения, связанных с учётом, анализом и обработкой информации об обслуживаемом парке компьютерной техники. Система позволит, на основе хранимой информации проводить анализ данных, и помогать в задачи планирования технического обслуживания и ремонта компьютеров. Также АИС позволит повысить качество управленческих решений руководства организации, за счет своевременного получения достоверной информации о состоянии дел в отделе компьютерного обеспечения.

База данных спроектирована с учетом реализации запросов различного типа по получению информации. При проектировании базы данных учитываются возможности выдачи бумажного отчета. Целями проектирования базы данных являются:

1. Эффективная структуризация информации, что позволяет сэкономить время и деньги;
2. Исключение или сведение к минимуму повторяющихся данных путем задания эффективной структуры;
3. Обеспечение всем пользователям быстрого доступа к информации базы данных;
4. Обеспечение расширения базы новыми данными;
5. Обеспечение целостности данных для того, чтобы база содержала только проверенную информацию;
6. Обеспечение базы данных интуитивно-понятным пользовательским интерфейсом.

Ни одна современная организация не может быть достаточно конкурентоспособной и эффективной, если в ней не проводятся усовершенствования и не реализуются новые идеи по автоматизации. К примеру, если установить на производстве несколько машин с числовым управлением (то есть таких, работой которых управляет компьютер), то можно достичь не только увеличения объемов производства, но и снижения цены на продукцию за счет сокращения штата сотрудников. Качество же товаров от такого перехода только улучшится, а благодаря сниженной цене они будет пользоваться большей популярностью на рынке. Для максимальной автоматизации производства на всех его этапах используются гибкие производственные системы, состоящие из множества элементов. Благодаря таким системам человек может практически не участвовать в процессе производства, осуществляя лишь контроль и управление.

Современные организации производят и накапливают огромные объемы данных. От того, насколько эффективно эта информация используется сотрудниками, руководителями, управляющими органами, зависит уровень развития страны в целом и каждого ее территориального субъекта в частности. Поэтому необходимость использования больших, и при этом еще постоянно растущих, объемов информации при решении задач, обуславливает сегодня создание информационных систем в организациях.

Актуальность развития информационных технологий подчеркивается президентом Д.А. Медведевым на заседании президиума Государственного совета «О реализации Стратегии развития информационного общества в Российской Федерации», проведенной 17 июля 2008 года: «…У нас на наших заседаниях президиума всегда рассматриваются наиболее актуальные вопросы развития нашей страны. К числу таковых относится вопрос развития информационного общества в Российской Федерации. Не буду говорить банальностей, очевидно, что в XXI веке главная ставка делается именно на развитие информационно-коммуникационных технологий. Этим всѐ сказано…»

**1 Характеристика предприятия**

В соответствии с постановлением Правительства РФ от 10 июля 2013 г. № 582 и приказом Федеральной службы по надзору в сфере образования и науки от 29 мая 2014 г. № 785

Полное наименование образовательной организации: государственное автономное профессиональное образовательное учреждение «Акбулакский политехнический техникум».

Сокращенное наименование: ГАПОУ «Акбулакский политехнический техникум».

Техникум создан в 1947 г. как училище механизации решением Чкаловского обкома №1167 от 06.08.1947 г.

На основании Постановления Совета Министров РСФСР №386 от 31.08.1984г. СПТУ №1 переименовано в ССПТУ № 51.

ССПТУ № 51 переименовано в  ПУ № 51 на основании приказа УПО № 27 -  09/101  от 10.11.1994 г.

ПУ № 51 переименовано в ГОУ НПО ПУ № 51 на основании приказа УПО № 01/11 – 180 от 08.05.2001г.

ГОУ НПО ПУ №51 переименовано в ГОУ НПО ПУ № 51 п. Акбулак Оренбургской области на основании приказа УПО от 05.05.2003 г. № 01/11 - 244

ГОУ НПО ПУ № 51 п. Акбулак Оренбургский области переименовано в ГАОУ НПО ПУ № 51 пос. Акбулака Оренбургской области на основании постановления правительства Оренбургской области от 20.12.2010 г. № 908 п.

ГАОУ НПО ПУ № 51 пос. Акбулака Оренбургской области переименовано в ГАОУ СПО «Акбулакский политехнический техникум» на основании постановления правительства Оренбургской области от 20.05.2013 № 383 - п.

Государственное автономное образовательное учреждение «Акбулакский политехнический техникум» переименовано постановлением Правительства Оренбургской области от 03.10.2014 № 751-п «О переименовании государственного автономного образовательного учреждения среднего профессионального образования «Акбулакский политехнический техникум» в государственное автономное профессиональное образовательное учреждение «Акбулакский политехнический техникум».

Основным видом деятельности организации является: образование профессиональное среднее. Расчет экономической эффективности представлен в Приложении А.

**2 Техническое задание**

Краткая характеристика объекта. Полное наименование АИС: Автоматизированная информационная система учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум».

Сокращённое наименование: АИС «Учет заявок».

Разработчик: Никифорова Анастасия .

Заказчик:ГАПОУ «Акбулакский политехнический техникум».

Настоящее техническое задание распространяется на разработку автоматизированной информационной системы учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». Основной целью разработки АИС является автоматизация учета заявок на ремонт компьютерной техники, а также учета оборудования. Актуальность данной задачи заключается в том, что процесс учета, хранения и анализа данных будут выполнятся в более подробной форме и гораздо быстрее, при этом риск потери сведений сводится к минимуму.

В настоящее время процесс накопления, обработки и использования знаний постоянно ускоряется. В связи с этим возникает необходимость использования современных информационных средств, которые значительно сокращают трудоёмкость процесса ведения и систематизации информации и увеличивают эффективность деятельности организации.

Основание для разработки. Система разрабатывается на основании учебного плана по специальности и договора на прохождение практики. Так же основанием для разработки послужило распоряжение директора организации, о создании автоматизированной информационной системы учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум».

Назначение и цели создаваемой автоматизированной системы. Система предназначена для автоматизации информационного процесса учета заявок пользователей на ремонт компьютерного оборудования.

Данная версия системы предназначена для внесения, хранения и выборки данных, позволяет снизить трудоемкость и увеличить скорость выполнения работы. С ростом числа используемой компьютерной техники в организации возникла проблема учёта, как компьютеров, так и отдельных комплектующих, а также проводимых технических осмотров и ремонтов. Хранить информацию и производить её выборку без специально разработанной автоматизированной информационной системы очень сложно. Технические осмотры и ремонт производятся несвоевременно, запасные детали не закупаются вовремя. Руководство организации не получает реальной картины по состоянию ИТ инфраструктуры.

Пользователем могут выступать сотрудники отдела компьютерного обеспечения.

В следующей версии программы возможна разработка web-портала, для самостоятельной регистрации заявок на обслуживание пользователями.

Требования к функциональным характеристикам. АИС должна представлять совокупность методических и программных средств выполнения следующих функций:

* соединение с базой данных;
* авторизация пользователей (на уровне СУБД);
* ведение справочников (добавление, удаление, редактирование записей таких справочников как: статус, тип обращения, тип оборудования, тип пользователя, производитель, пользователь);
* ведение учета заявок на ремонт (добавление, удаление, редактирование заявок на ремонт: обращение с параметрами, ремонт, манипуляция);
* учет компьютерного оборудования (добавление, удаление, редактирование данных об оборудования: справочник оборудования, производитель);
* простой и наглядный пользовательский интерфейс;
* формирование отчетов (список заявок, список сотрудников, список оборудования);
* обеспечить многопользовательский доступ к программному средству.

Методическое обеспечение должно быть реализовано в пользовательском интерфейсе системы, который должен предполагать выбор объекта системы, ввод, изменение, просмотр и хранение данных, решение задачи статистического анализа. Программное средство должно иметь возможность к модернизации и последующей модификации.

Требования к надежности. Предусмотреть контроль вводимой информации и блокировку некорректных действий пользователя при работе с системой. Обеспечить целостность информации, хранящейся в базе данных.

Требования к составу и параметрам технических средств. Система должна работать на x86\64 компьютере. Минимальная конфигурация представлена в таблице 2.

Таблица 2 – Характеристики компьютеров

|  |  |  |
| --- | --- | --- |
| № | Наименование | Характеристика |
| 1 | Процессор | Не менее 2 ядер с частотой не менее 1,7 Ггц |
| 2 | ОЗУ | DDR 3, не менее 4 Гб |
| 3 | Жесткий диск | Не менее 2 Гб свободного места |
| 4 | Мониторы | От 24 дюймов, с разрешение не менее 1980 на 1080 |

Требования к информационной и программной совместимости. Система должна работать под управлением операционной системы Windows 7 и выше, СУБД MS SQL Server 2012 или новее.

Microsoft SQL Server - одна из наиболее мощных систем работы с базами данных в архитектуре «клиент-сервер». Особенность системы - работа сервера только в операционных системах ряда Microsoft Windows, при этом клиентская часть может взаимодействовать с сервером из других операционных систем. Рекомендуемая файловая система для SQL Server - NTFS, хотя возможна работа и в других [15, с. 56].

В своем составе система имеет средства создания баз данных, работы с информацией баз данных, перенесения данных из других систем и в другие системы, резервного копирования и восстановления данных, развитую систему транзакций, систему репликации данных, реляционную подсистему для анализа, оптимизации и выполнения запросов клиентов, систему безопасности для управления правами доступа к объектам базы данных и пр. Система не содержит средств разработки клиентских приложений.

Существует несколько способов подключения к SQL Server. Чаще всего используют четыре варианта:

* подключение по OLE DB — набор интерфейсов, основанных на СОМ, которые позволяют приложениям обращаться к данным, хранимым в разных источниках информации или хранилищах данных с помощью унифицированного доступа. Этот способ является наиболее рекомендуемым и современным вариантом;
* подключение по Open DataBase Connectivity (ODBC) — интерфейс взаимодействия приложений с системами управления базами данных (СУБД);
* подключение с использованием BDE (Borland Database Engine) — доступ к базам данных из C++ Builder, Delphi;
* подключение по JDBC, который позволяет приложениям под управлением Java соединяться с Microsoft SQL Server.

Всё используемое ПО работает под управление ОС Windows 7. В этой операционной системе реализована поддержка Unicode 5.1. Панель поиска Instant Search теперь распознаёт больше языков. Данная ОС обладает поддержкой мультитач-управления.

Все версии ОС включают 50 новых шрифтов. Существующие шрифты доработаны для корректного отображения всех символов. Windows 7 — первая версия Windows, которая включает больше шрифтов для отображения нелатинских символов, чем для отображения латинских. Панель управления шрифтами также подверглась улучшению — по умолчанию, в ней будут отображаться только те шрифты, раскладка для которых установлена в системе.

Windows 7 поддерживает псевдонимы для папок на внутреннем уровне. К примеру, папка Program Files в некоторых локализованных версиях Windows была переведена и отображалась с переведённым именем, однако на уровне файловой системы оставалась англоязычной. Также в систему (кроме версии Windows 7 Starter и Windows 7 Home Basiс) встроено около 120 фоновых рисунков, уникальных для каждой страны и языковой версии. Так, русская версия включает тему «Россия» с шестью уникальными обоями высокого разрешения.

Дополнительным преимуществом Windows 7 можно считать более тесную интеграцию с производителями драйверов. Большинство драйверов определяются автоматически, при этом в 90 % случаев сохраняется обратная совместимость с драйверами для Windows Vista.

В Windows 7 была также улучшена совместимость со старыми приложениями, некоторые из которых было невозможно запустить на Windows Vista. Особенно это касается старых игр, разработанных под Windows XP. Также в Windows 7 появился режим Windows XP Mode, позволяющий запускать старые приложения в виртуальной машине Windows XP, что обеспечивает практически полную поддержку старых приложений [16, с. 45].

Новая, 11-я версия DirectX, впервые выпущенная именно в составе этой ОС, имеет следующие улучшения: добавлена поддержка новых вычислительных шейдеров, возможность многопоточного рендеринга, улучшена тесселяция, появились новые алгоритмы компрессии текстур и др.

Сетевая технология Branch Cache позволяет кешировать содержимое интернет-трафика. Если пользователю в локальной сети потребуется файл, который уже был загружен кем-то из пользователей его сети, — он сможет получить его из локального кэш-хранилища, а не использовать канал с ограниченной пропускной способностью. Технология рассчитана на крупные сети и предлагается для внедрения на предприятиях в составе Корпоративной и Максимальной версий ОС.

Windows AIK, с помощью которого можно создавать образ с любыми конфигурациями и настройками. Таким образом можно в Windows 7 Добавить поддержку USB 3.0, Bluetooth 4.0, DirectX 11.1 и NET.Framework 4.5 по умолчанию.

Требования к программной документации. Разрабатываемая система должна включать справочную информацию о работе системы и подсказки пользователю.

В состав сопровождающей документации должны входить:

* пояснительная записка;
* руководство пользователя;
* руководство программиста.

Требования к составу и содержание работ по созданию системы. Формирование требований к АИС: Обследование объекта и обоснование необходимости создания АИС. Формирование требований пользователя к АИС. Оформление отчета о выполненной работе и заявки на разработку АИС (тактико-технического задания)

Разработка концепции АИС: Изучение объекта. Проведение необходимых научно-исследовательских работ. Разработка вариантов концепции АИС, удовлетворяющих требованиям пользователя Оформление отчета о выполненной работе

Техническое задание: Разработка и утверждение технического задания на создание АИС.

Эскизный проект: Разработка предварительных проектных решений по системе и ее частям Разработка документации на АИС и ее части

Технический проект: Разработка проектных решений по системе и ее частям. Разработка документации на АИС и ее части. Разработка и оформление документации на поставку изделий для комплектования АИС и (или) технических требований (технических заданий) на их разработку. Разработка заданий на проектирование в смежных частях. Проекта объекта автоматизации

Рабочая документация: Разработка рабочей документации на систему и ее части. Разработка или адаптация программ

Ввод действий: Подготовка объекта автоматизации к вводу АИС в действие. Подготовка персонала. Комплектация АИС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями.) Проведение предварительных испытаний.

Сопровождение АИС: Выполнение работ в соответствии с гарантийными обязательствами. Послегарантийное обслуживание.

Порядок контроля и приемки системы. Испытания АИС проводят на стадии «Ввода в действие» с целью проверки соответствия создаваемой АИС требованиям технического задания (ТЗ). Испытания АИС представляют собой процесс проверки выполнения заданных функций системы, определения и проверки соответствия требованиям ТЗ количественных и качественных характеристик системы, выявления и устранения недостатков в действиях системы, в разработанной документации. Для АИС устанавливают следующие основные виды испытаний.

Предварительные. В зависимости от взаимосвязей, испытываемых в АИС объектов испытания могут быть автономные или комплексные.

Автономные испытания охватывают части АИС. Их проводят по мере готовности частей АИС к сдаче в опытную эксплуатацию. В программе автономных испытаний указывают:

* перечень функций, подлежащих испытаниям;
* описание взаимосвязей объекта испытаний с другими частями АИС;
* условия, порядок и методы проведения испытаний и обработки результатов;
* критерии приемки частей по результатам испытаний.

В качестве исходной информации для теста рекомендуется использовать фрагмент реальной информации организации-заказчика в объеме, достаточном для обеспечения необходимой достоверности испытаний. Результаты автономных испытании частей АИС следует фиксировать в протоколах испытаний. Протокол должен содержать заключение о возможности (невозможности) допуска части АИС к комплексным испытаниям. В случае, если проведенные автономные испытания будут признаны недостаточными, либо будет выявлено нарушение требований регламентирующих документов по составу или содержанию документации, указанная часть АИС может быть возвращена на доработку и назначен новый срок испытаний [17, с. 43].

Комплексные испытания проводят для групп, взаимосвязанных частей АИС или для АИС в целом.

В программе комплексных испытаний АИС или частей АИС указывают:

* перечень объектов испытания;
* состав предъявляемой документации;
* описание проверяемых взаимосвязей между объектами испытаний;
* очередность испытаний частей АИС;
* порядок и методы испытаний, в том числе состав программных средств и оборудования, необходимых для проведения испытаний, включая специальные стенды и полигоны.

При комплексных испытаниях допускается использовать в качестве исходной информацию, полученную на автономных испытаниях частей АИС.

Протокол комплексных испытаний должен содержать заключение о возможности (невозможности) приемки АИС в опытную эксплуатацию, а также перечень необходимых доработок и рекомендуемые сроки их выполнения. После устранения недостатков проводят повторные комплексные испытания в необходимом объеме.

Для планирования проведения всех видов испытаний разрабатывают документ «Программа и методика испытаний». Разработчик документа устанавливается в договоре или ТЗ. Программа и методика испытаний должны устанавливать необходимый и достаточный объем испытаний, обеспечивающий заданную достоверность получаемых результатов.

Опытную эксплуатацию АИС проводят с целью определения фактических значений количественных и качественных характеристик АИС и готовности персонала к работе в условиях функционирования АИС, определения фактической эффективности АИС, корректировке (при необходимости) документации.

Приемочные испытания АИС проводят для определения, соответствия АС техническому заданию, оценки качества опытной эксплуатации и решения вопроса о возможности приемки АИС в постоянную эксплуатацию. Приемочным испытаниям АИС должна предшествовать ее опытная эксплуатация на объекте.

При испытаниях АИС проверяют:

* качество выполнения комплексом программных и технических средств автоматических функций во всех режимах функционирования АИС согласно ТЗ на создание АИС;
* знание персоналом эксплуатационной документации наличие у него навыков, необходимых для выполнения установленных функций во всех режимах функционирования АИС, согласно ТЗ на создание АИС;
* полноту содержащихся в эксплуатационной документации указании персоналу по выполнению им функций во всех режимах функционирования АС, согласно ТЗ на создание АИС;
* количественные и (или) качественные характеристики выполнения автоматических и автоматизированных функций АИС в соответствии с ТЗ;
* другие свойства АИС, которым она должна соответствовать по ТЗ.

Испытания АИС следует проводить на объекте заказчика. По согласованию между заказчиком и разработчиком предварительные испытания и приемку программных средств АИС допускается проводить на технических средствах разработчика при создании условий получения достоверных результатов испытаний.

Допускается последовательное проведение испытаний и сдача частей АИС в опытную и постоянную эксплуатацию при соблюдении установленной в ТЗ очередности ввода АИС в действие.

Приемочные испытания следует проводить на функционирующем объекте. Приемочные испытания в первую очередь должны включать проверку:

* полноты и качества реализации функций при штатных, предельных, критических значениях параметров объекта автоматизации и в других условиях функционирования АИС, указанных в ТЗ;
* выполнения каждого требования, относящегося к интерфейсу системы;
* работы персонала в диалоговом режиме;
* средств и методов восстановления работоспособности АИС после отказов;
* комплектности и качества эксплуатационной документации.

Проверку полноты и качества выполнения функций АИС рекомендуется проводить в два этапа. На первом этапе проводят испытания отдельных функций (задач, комплексов задач). При этом проверяют выполнение требований ТЗ к функциям (задачам, комплексам задач). На втором этапе проводят проверку взаимодействия задач в системе и выполнение требований ТЗ к системе в целом. По согласованию с заказчиком проверка задач в зависимости от их специфики может проводиться автономно или в составе комплекса. Объединение задач при проверке в комплексах целесообразно проводить с учетом общности используемой информации и внутренних связей [18, с. 13-20].

Проверку работы персонала в диалоговом режиме проводят с учетом полноты и качества выполнения функций системы в целом.

Проверке подлежит:

* полнота сообщений, директив, запросов, доступных оператору и их достаточность для эксплуатации системы;
* сложность процедур диалога, возможность работы персонала без специальной подготовки;
* реакция системы и ее частей па ошибки оператора, средства сервиса.
* проверка средств восстановления работоспособности АС после отказов ЭВМ должна включать:
* проверку наличия в эксплуатационной документации рекомендаций по восстановлению работоспособности и полноту их описания;
* практическую выполнимость рекомендованных процедур;
* работоспособность средств автоматического восстановления функций (при их наличии).

Проверку комплектности и качества эксплуатационной документации следует проводить путем анализа документации на соответствие требованиям нормативно-технических документов ТЗ.

Результаты испытаний объектов, предусмотренных программой, фиксируют в протоколах, содержащих следующие разделы:

* назначение испытаний и номер раздела требований ТЗ на АИС, по которому проводят испытание;
* состав технических и программных средств, используемых при испытаниях;
* указание методик, в соответствии с которыми проводились испытания, обработка и оценка результатов;
* условия проведения испытаний и характеристики исходных данных;
* средства хранения и условия доступа к конечной, тестирующей программе;
* обобщенные результаты испытаний;
* выводы о результатах испытаний и соответствии созданной системы или ее частей определенному разделу требований ТЗ на АИС.

Протоколы испытаний объектов по всей программе обобщают в едином протоколе, на основании которого делают заключение о соответствии системы требованиям ТЗ на АИС и возможности оформления акта приемки АИС в постоянную эксплуатацию.

Работу завершают оформлением акта о приемке АИС в постоянную эксплуатацию.

Требования к составу и содержанию работ по подготовки объекта автоматизации к вводу системы в действие.

При создании и внедрении автоматизированной системы изменяется принцип работы программ.

При внедрении автоматизированной информационной системы учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». Сотрудники должны научиться работать с ней. При необходимости провести обучение сотрудников, для работы с данной системой. Необходимо оборудовать место для работы пользователя с системой, при необходимости отрегулировать соответствующие показатели.

На момент внедрения системы, на ПК должны быть установлены: операционная система, программное средство и все технические средства, необходимые для безошибочного функционирования программы-принтер.

**3 Информационное обеспечение**

В базах данных содержится информация об объектах, существующих в реальном мире. Все эти объекты являются материальными, обладающим некоторыми свойствами и присущим им поведением. При создании программного продукта, свойства объектов хранятся в базе данных, а их поведение реализуется в приложении посредством методов – процедур, которые представляют собой отклик на события инициируемых пользователем.

Инфологическая модель является проблемно-ориентированной и системно-независимой, т.е. не зависимой от конкретной СУБД, операционной системы и аппаратного обеспечения ЭВМ.

Основным требованием к инфологической модели, вытекающим из ее назначения, является требование адекватного отображения предметной области. Инфологическая модель должна быть непротиворечивой. Она является единым интегрированным описанием предметной области и отражает взгляды и потребности всех пользователей системы [20, с. 34-37].

Инфологическая модель представляет информационные потоки, сущности и связи данной предметной области. Она может быть представлена в виде ER-модели и реляционной схемы. Мы будем представлять инфологическую модель в виде ER-модели. Она должна обладать свойством легкой расширяемости, обеспечивающим ввод новых данных без изменений ранее определенных. Центральной компонентой инфологической модели является описание объектов предметной области и связей между ними (ER-модель).

Цель инфологического моделирования – обеспечение наиболее естественных для человека способов сбора и представления той информации, которую предполагается хранить в создаваемой базе данных [21, с. 5-8]. Наглядно информационная модель представлена в приложении Г на рисунке Г.1.

Основными конструктивными элементами инфологических моделей являются сущности, связи между ними, идентификаторы (ключи) и свойства (атрибуты). Выделим сущности, которые будут использованы при построении базы данных, и определим характеризующие их атрибуты:

1. Сущность «Статус»;
   1. Код;
   2. Название;
2. Сущность «Ремонт»;
   1. Код;
   2. Название;
3. Сущность «Манипуляции»;
   1. Код;
   2. Название;
4. Сущность «Запасная часть»;
   1. Код;
   2. Название;
5. Сущность «Производитель»;
   1. Код;
   2. Название;
   3. Краткое название;
6. Сущность «Тип пользователя»;
   1. Код;
   2. Название;
7. Сущность «Тип оборудования»;
   1. Код;
   2. Название;
8. Сущность «Пользователь»;
   1. Код;
   2. Фамилия;
   3. Имя;
   4. Отчество;
   5. Электронная почта;
   6. Телефон;
9. Сущность «Тип обращения»;
   1. Код;
   2. Название;
10. Сущность «Справочник оборудования»;
    1. Код;
    2. Название;
    3. Год выпуска;
    4. Модель;
    5. Серийный номер;
    6. Инвентарный номер;
11. Сущность «Обращение»;
    1. Код;
    2. Номер;
    3. Дата\время регистрации;
    4. Дата\время разрешения;
    5. Описание;
    6. Отложено до.

Даталогическая модель является моделью логического уровня и представляет собой отображение логических связей между элементами данных безотносительно к их содержанию и среде хранения. Эта модель строится в терминах информационных единиц, допустимых в той конкретной СУБД, в среде которой мы проектируем базу данных. Этап создания даталогической модели называется даталогическим проектированием. Описание логической структуры базы данных на языке СУБД называется схемой [22, с. 34-37].

Хотя даталогическое проектирование является логической структурой базы данных, на него влияют возможности физической организации данных, представляемые конкретной СУБД. Поэтому знание особенностей физической организации данных является полезным при проектировании логической структуры.

Логическая структура базы данных, а также сама заполненная данными база данных является отображением реальной предметной области. Поэтому на выбор проектных решений самое непосредственное влияние оказывает специфика отображаемой предметной области, отраженная в инфологической модели.

Для реляционной базы данных проектирование логической структуры заключается в том, чтобы разбить всю информацию по отношениям, а также определить состав атрибутов для каждого из этих файлов.

Все данные и другая информация СУБД хранятся на магнитных дисках в дисковых файлах. Файл данных представляет собой таблицу, каждая строка которой содержит некоторые сведения об описываемом объекте. Все записи базы данных имеют идентичную, заданную пользователем структуру и размеры.

Для привязки даталогической модели к среде хранения используется модель данных физического уровня. Эта модель определяет используемые запоминающие устройства, способы физической организации данных в среде хранения. Модель физического уровня также строится с учетом возможности, представляемых СУБД. Описание физической структуры базы данных называется схемой хранения. Соответствующий этап проектирования базы данных называется физическим проектированием.

Всякая реляционная база данных должна быть нормализована. Процесс нормализации имеет своей целью устранение избыточности данных и заключается в приведении к нормальной форме Бойса-Кодда. Нормализацией называется формальная процедура, в ходе которой атрибуты данных группируются в таблицы, а таблицы группируются в базу данных.

Единицей, хранящейся в БД, информации является таблица. Каждая таблица представляет собой совокупность строк и столбцов, где строки соответствуют экземпляру, а столбцы - атрибутам. В терминах БД столбцы таблицы называются полями, а ее строки - записями.

Нормализация выполняется поэтапно. Первые три шага были описаны доктором Э.Ф. Коддом в статье "Дальнейшая нормализация реляционной модели базы данных" в 1972г.

Первая нормальная форма (1НФ). Для нее требуется, чтобы таблица была плоской и не содержала повторяющихся групп. У плоской таблицы есть только две характеристики - длина (количество записей или строк) и ширина (количество полей или столбцов). Такая таблица не должна содержать ячеек, включающих несколько значений.

Никакую из систем управления базами данных (СУБД) не удовлетворяет только 1НФ, так как в этом случае необходимо определить большое число полей, многие из которых остаются в основном пустыми. Избыточные данные могут послужить причиной проблем целостности и снижение эффективности при внесении изменений, поэтому подобных решений при проектировании баз данных необходимо избегать.

Вторая нормальная форма (2НФ). Для 2НФ требуется, чтобы все поля таблицы зависели от первичного ключа, то есть, чтобы первичный ключ однозначно определял запись и не был избыточен. В каждой таблице БД может существовать первичный ключ - поле или набор полей, однозначно идентифицирующий запись. Значение первичного ключа в таблице БД должно быть уникальным, т.е. в таблице не должно существовать двух и более записей с одинаковым значением первичного ключа. Те поля, которые зависят только от части первичного ключа, должны быть выделены в составе отдельных таблиц. 2НФ позволяет удалить большую часть повторяющихся данных, которые часто остаются после первого этапа нормализации.

Для третьей нормальной формы (ЗНФ) требуется, чтобы все не ключевые столбцы таблицы зависели от первичного ключа таблицы, но были независимы друг от друга. Для этого требуется, чтобы таблицы были приведены к 1НФ и 2НФ.

Введем определение детерминанта. Детерминант — это любой атрибут, от которого полностью функционально зависит некоторый другой атрибут. Введя данное понятие, можно дать определение НФБК. Отношение будет находиться в НФБК, если оно находится в 3НФ и каж­дый детерминант является потенциальным ключом.

Отношение находится в четвертой нормальной форме (4НФ), если оно находится в НФБК и не имеет много­значных зависимостей. Отношение находится в пятой нормальной форме (5НФ) только в том случае, когда любая зависимость соединения в отношении следует из существования некоторого возможного ключа в данном отношении. Отношение находится в доменно-ключевой нор­мальной форме (ДКНФ), если каждое ограничение, накладываемое на это отношение, яв­ляется логическим следствием определения доменов и ключей.

Для большинства существующих СУБД необходимо представить проект базы данных в ЗНФ, так как этого вполне достаточно практически для всех обычных приложений. При разработке исключительно больших систем, когда необходимо максимальное сокращение объемов, хранимых данных, желательно произвести дальнейшую нормализацию. Но в этом случае можно столкнуться с недостатками нормализации. Поскольку порог человеческого восприятия не позволяет одновременно анализировать большое число объектов с учетом их взаимосвязей, можно утверждать, что с увеличением числа нормализованных таблиц уменьшается целостное восприятие базы данных как системы взаимосвязанных данных. Другим недостатком нормализованной базы данных является необходимость считывать связанные данные из нескольких таблиц при выполнении одного запроса [23, с. 55-61].

Данные в таблицах MS SQL Server сохраняются в определенном формате, который называется типом данных. Типы данных могут быть классифицированы по четырем категориям: числовые (numeric), символьные (character), даты (date) и Logical (логические).

Числовые данные включают в себя все числа, начиная с целых вплоть до чисел двойной точности с плавающей точкой. Символьные данные содержат строки текста. Даты используются для хранения дат и времени. В нашей базе данных используются следующие типы данных: числовой, текстовый, счётчик, дата/время.

Структура записей таблиц баз данных приведены в таблицах 3-13.

Таблица 3 – Описание таблицы «Статус»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойство | Ключи | Физические  характеристики | Логические  ограничения | Процессы со  свойством |
| код | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название |  | Символьный,  50 символов |  | В, К, П, У |
| кр. название |  | Символьный,  15 символов |  | В, К, П, У |

Таблица 4 – Описание таблицы «Ремонт»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойство | Ключи | Физические  характеристики | Логические  ограничения | Процессы со  свойством |
| Код | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название |  | Символьный,  50 символов |  | В, К, П, У |
| манипуляция | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| зап. часть | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| обращение | Fk | Длинное целое,  4 байта |  | В, К, П, У |

Таблица 5 – Описание таблицы «Пользователь»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойство | Ключи | Физические  характеристики | Логические  ограничения | Процессы со  свойством |
| код | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| фамилия |  | Символьный,  50 символов |  | В, К, П, У |
| имя |  | Символьный,  50 символов |  | В, К, П, У |
| отчество |  | Символьный,  50 символов |  | В, К, П, У |
| эл. почта |  | Символьный,  50 символов |  | В, К, П, У |
| телефон | Fk | Символьный,  50 символов |  | В, К, П, У |
| тип пользователя | Fk | Длинное целое,  4 байта |  | В, К, П, У |

Таблица 6 – Описание таблицы «Манипуляция»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойство | Ключи | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название |  | Символьный,  50 символов |  | В, К, П, У |

Таблица 7 – Описание таблицы «Запасная часть»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойство | Ключи | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название |  | Символьный,  50 символов |  | В, К, П, У |
| производитель | Fk | Длинное целое,  4 байта |  | В, К, П, У |

Таблица 8 – Описание таблицы «Тип оборудования»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойство | Ключи | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название |  | Символьный,  50 символов |  | В, К, П, У |

Таблица 9 – Описание таблицы «Тип пользователя»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Свойство | Ключи | | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название | |  | Символьный,  50 символов |  | В, К, П, У |

Таблица 10 – Описание таблицы «Производитель»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Свойство | Ключи | | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название | |  | Символьный,  50 символов |  | В, К, П, У |
| кр. название | |  | Символьный,  15 символов |  | В, К, П, У |

Таблица 11 – Описание таблицы «Тип обращения»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Свойство | Ключи | | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название | |  | Символьный,  50 символов |  | В, К, П, У |

Таблица 12 – Описание таблицы «Справочник оборудования»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Свойство | Ключи | | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| код | | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| название | |  | Символьный,  50 символов |  | В, К, П, У |
| год выпуска | |  | Длинное целое,  4 байта |  | В, К, П, У |
| модель | |  | Символьный,  30 символов |  | В, К, П, У |
| серийный номер | |  | Символьный,  20 символов |  | В, К, П, У |
| инвентарный номер | |  | Символьный,  20 символов |  | В, К, П, У |
| тип оборудования | | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| производитель | | Fk | Длинное целое,  4 байта |  | В, К, П, У |

Таблица 13 – Описание таблицы «Обращения»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Свойство | Ключи | | Физические  характеристики | Логические  ограничения | Процессы со свойством |
| 1 | 2 | | 3 | 4 | 5 |
| код | | Uk, Pk | Длинное целое,  4 байта | Not Null | Г, П, У |
| номер | |  | Длинное целое,  4 байта |  | В, К, П, У |
| дата\время регистрации | |  | Дата\время |  | В, К, П, У |
| дата\время разрешения | |  | Дата\время |  | В, К, П, У |
| описание | |  | Символьный,  150 символов |  | В, К, П, У |
| отложено до | |  | Дата\время |  | В, К, П, У |

Продолжение таблицы 13

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| производитель | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| статус | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| тип обращения | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| справочник об-я | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| Пользователь 1 | Fk | Длинное целое,  4 байта |  | В, К, П, У |
| Пользователь 2 | Fk | Длинное целое,  4 байта |  | В, К, П, У |

В таблицах используются следующие сокращения. Ключи – сокращения: «Pk» (Primary key) – первичный ключ, «Uk» (Unique key) – уникальный ключ, «FK» (Foreign key) – внешний ключ. Процессы со свойством – сокращения: «Г» - генерация, «В» – ввод, «П» – просмотр, «К» – корректировка, «У» – удаление.

Выявленные классы объектов, присущие заданной предметной области, с отображением связей между ними представлены в таблице 14.

Таблица 14 – Связи между классами объектов

| Классы  Объектов | | Опциональ-ность связи | | Имя связи со  стороны | | Тип связи со стороны | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| главн. | подч. | глав. | подч. | главн. | подч. | главн. | подч. |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Тип оборудования | Справочник оборудования | М.б. | Д.б. | соответствует | относится | 1 | М |
| Производитель | Справочник оборудования | М.б. | Д.б. | соответствует | относится | 1 | М |
| Справочник оборудования | Обращения | М.б. | Д.б. | соответствует | относится | 1 | М |
| Справочник ПО | Обращения | М.б. | Д.б. | соответствует | относится | 1 | М |
| Тип пользователя | Пользователь | М.б. | Д.б. | соот-ветствует | относится | 1 | М |
| Пользователь | Обращение | М.б. | Д.б. | разрешил | относится | 1 | М |
| Пользователь | Обращение | М.б. | Д.б. | зарегистрировал | относится | 1 | М |

Продолжение таблицы 14

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Статус | Обращение | М.б. | Д.б. | соот-ветствует | относится | 1 | М |
| Производитель | Запасная часть | М.б. | Д.б. | соот-ветствует | относится | 1 | М |
| Тип обращения | Обращение | М.б. | Д.б. | соот-ветствует | относится | 1 | М |
| Запасная часть | Ремонт | М.б. | Д.б. | соот-ветствует | относится | 1 | М |
| Манипуляция | Ремонт | М.б. | Д.б. | соот-ветствует | относится | 1 | М |

Описание групп пользователей АИС и уровней их доступа представлено в таблице 15.

Таблица 15 - Уровни доступа пользователей

| Класс объектов/ свойства | Пользователи | | |
| --- | --- | --- | --- |
| Администратор | Нач. отдела | Инженеры отдела |
| 1 | 2 | 3 | 4 |
| Тип пользователя |  |  |  |
| код | RUID | RI | R |
| название | RUID | RUI | R |
| Пользователь |  |  |  |
| Фамилия | RUID | RI | RI |
| Имя | RUID | RUI | RUI |
| Отчество | RUID | RUI | RUI |
| Эл. почта | RUID | RUI | RUI |
| Телефон | RUID | RUI | RUI |
| Контакты | RUID | RUI | RUI |
| Тип оборудования |  |  |  |
| код | RUID | RI | R |
| название | RUID | RUI | R |
| Производитель |  |  |  |
| код | RUID | RI | RI |
| название | RUID | RUI | RUI |
| краткое название | RUID | RUI | RUI |
| Статус |  |  |  |
| код | RUID | RI | R |
| название | RUID | RUI | R |
| краткое название | RUID | RUI | R |
| Тип обращения |  |  |  |
| код | RUI | RI | R |
| название | RUID | RUI | R |
| краткое название | RUID | RUI | R |
| Справочник оборудования |  |  |  |
| Код | RUI | RI | RI |
| Название | RUID | RUI | RUI |
| Год выпуска | RUID | RUI | RUI |
| Модель | RUID | RUI | RUI |
| Серийный номер | RUID | RUI | RUI |
| Инвентарный номер | RUID | RUI | RUI |
| Зап. часть |  |  |  |
| Код | RUI | RI | RI |
| Название | RUID | RUI | RUI |
| Манипуляция |  |  |  |
| Код | RUI | RI | R |
| Название | RUID | RUI | R |
| Ремонт |  |  |  |
| Код | RUI | RI | RI |
| Название | RUID | RUI | RI |

**4 Описание автоматизированных функций**

Важной стороной проектирования является описание функционального назначения программного средства, которое позволяет определить масштаб разработки. Разработанная функциональная схема АИС представлена на рисунке Д.1 в приложении Д.

Функциональная схема или схема данных – схема взаимодействия компонентов программного обеспечения с описанием информационных потоков, состава данных в потоках и указанием используемых файлов и устройств.

Функциональная схема программного продукта строится с целью однозначного понимания всех функций, выполняемых данной программой. В большинстве случаев функциональная спецификация формулируется на естественном языке при помощи специальных объектов и утверждений, конкретно описывающих функции программного средства.

Авторизация пользователя – процесс предоставления пользователю прав и проверка их наличия. Пользователю необходимо ввести имя пользователя и пароль. В случае успешной авторизации пользователь получает возможность работать с программным средством в пределах выделенных ему прав, при неудаче – выдается сообщение об ошибке и приложение предлагает совершить еще одну попытку ввода. На входе – база данных и набор данных. На выходе – база данных, сообщение об ошибке.

Ведение справочных данных – процесс ввода и редактирования справочной информации. На входе – база данных и новый набор данных. На выходе – база данных, данные на экране. Ведение справочных данных доступно администратору программы и начальнику отдела компьютерного обеспечения. Это такие категории данных как: статус, тип обращения, тип оборудования, тип пользователя, производитель, пользователь.

Ведение учета заявок на ремонт – функция, позволяющая с помощью сотрудника отдела фиксировать заявки пользователей, по ступающие по различным каналам. На входе – база данных и набор данных. На выходе – база данных, данные на экране. Ведение данных о заявках доступно всем сотрудникам отдела компьютерного обеспечения.

Учет компьютерного оборудования – функция, позволяющая сотрудникам отдела вести учет информации об имеющемся компьютерном оборудовании. На входе – база данных и набор данных. На выходе – база данных, данные на экране. Ведение данных об оборудовании доступно всем сотрудникам отдела компьютерного обеспечения.

Формирование отчетов – функция, позволяющая из исходных данных составить документ и отправить его на печать или вывести на экран. На входе – база данных. На выходе – результат формирования отчета или печатный вид документа. Формирование выходной документации доступно начальнику отдела. Формируются такие документы как: список заявок, список сотрудников, список оборудования.

**5 Руководство программиста**

Целью данной системы является учет заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». С помощью которой упростится поиск процесс учета заявок на ремонт техники, поиск информации и работа с документами, как для специалистов, так и для других сотрудников.

Автоматизированная информационная система реализует такие функции как:

* авторизация пользователей (на уровне СУБД);
* ведение справочников (добавление, удаление, редактирование записей таких справочников как: статус, тип обращения, тип оборудования, тип пользователя, производитель, пользователь);
* ведение учета заявок на ремонт (добавление, удаление, редактирование заявок на ремонт: обращение с параметрами, ремонт, манипуляция);
* учет компьютерного оборудования (добавление, удаление, редактирование данных об оборудования: справочник оборудования, производитель);
* простой и наглядный пользовательский интерфейс;
* формирование отчетов (список заявок, список сотрудников, список оборудования).

Для запуска данной АИС необходим пакет обновлений Net. Framework 4.5 для windows 7. Для отображения и наглядного представления программы необходима ОС WINDOWS 7 и выше.

Для работы программы также необходимо наличие технических средств, которые обладают следующими характеристиками, представленными в таблице 19.

Таблица 19 – Характеристика компьютера

|  |  |  |
| --- | --- | --- |
| № | Наименование | Характеристика |
| 1 | Процессор | Не менее 2 ядер с частотой не менее 2,2 Ггц |
| 2 | ОЗУ | DDR 3, не менее 4 Гб |
| 3 | Жесткий диск | 500 – 1000 Гб, не менее 2 Гб. Свободного пространства для приложения и не менее 5 Гб для сервера СУБД |
| 4 | Мониторы | От 24 дюймов, разрешение не менее FullHD 1980 на 1080 |

Для установки программы необходимо распаковать архив в папку «mat» на диск С:\. Данная папка содержит файл настроек подключения к серверу базы данных и исполняемый файл АИС. Для нормального функционирования сетевой версии программы необходимо что бы был запущен сервер СУБД.

При создании приложения базы данных часто возникает необходимость разграничить права пользователей для работы с данными. Кроме этого, ограничение доступа к базе, выполняет, по сути, еще и функцию защиты данных от несанкционированного доступа на уровне СУБД.

Процесс создания пользователей в MS SQL Server включает два этапа:

* Создание имени входа на SQL Server. Данное имя необходимо, для того чтобы предоставить пользователю возможность подключиться к экземпляру SQL Server;
* Создание пользователя базы данных. В данном случае мы уже предоставляем пользователю разрешения на объекты базы данных.

Прежде чем приступать к созданию имени входа на SQL сервер необходимо определиться с методом аутентификации. Существует два варианта:

1. Проверка подлинности Windows – это когда имя входа может идентифицировать пользователя как учетную запись Windows или как члена группы Windows (в том числе и доменные учетные записи, и группы);
2. Проверка подлинности SQL Server. В данном случае имя входа существует только в SQL Server.

Запускаем Management Studio, затем в обозревателе объектов находим пункт «Безопасность», раскрываем его плюсиком, кликаем правой кнопкой мыши по пункту «Имена входа» и выбираем пункт «Создать имя входа».

Для того чтобы создать имя входа на языке Transact-SQL необходимо в Management Studio открыть редактор запросов и выполнить следующую инструкцию (она делает ровно то же самое, что и наши действия выше в графическом интерфейсе Management Studio). Пример кода представлен на рисунке 5.

CREATE LOGIN [TestLogin]

WITH PASSWORD=N'Pa$$w0rd',

DEFAULT\_DATABASE=[Test],

DEFAULT\_LANGUAGE=[русский],

CHECK\_EXPIRATION=OFF,

CHECK\_POLICY=ON

GO

Рисунок 5 - Создание имени входа с использованием языка Transact-SQL

После установки необходимо произвести проверку программы. Проверим работу справочников. Выберите в главном меню «Справочники»\«Статус обращений». Добавим одну запись. В списке появился введённая запись.

Далее проверим удаление пользователей. Удалим введённую ранее запись. В главном меню выберем «Справочники»\«Пользователь». После этого выберем запись которую необходимо удалить, и нажмем на кнопку «Удалить». В появившемся окне нажмите «Да». В списке на форме исчезла запись.

Если все вышеописанные действия выполняются, то программа работает правильно. Для более подробного тестирования см. пункт «Тестирование и отладка»

При настройке и проверки программы возможно появление следующих сообщений (см. таблица 20):

Таблица 20 - Сообщения системному программисту

|  |  |
| --- | --- |
| Текст сообщений | Причины возникновения и устранение |
| Несоответствие типов | Переменная или ее свойство неправильного типа. Например, переменная, которая требует целое число. Поэтому необходимо выполнять присвоение только между совместимыми типами данных. |
| Не достаточно памяти | Загружено слишком много драйверов устройств. Для этого удалить ненужные драйверы.  Открыто слишком много документов, приложений. Для этого закрыть ненужные из них |
| Ошибка ввода-вывода | Возникают ошибки ввода-вывода при использовании принтера или дисковода. Убедитесь в правильной работе и повторите операции |
| Запрос на добавление не смог записать все данные | Возникает при добавлении новых данных уже содержащихся среди полей таблиц. |

**6 Руководство пользователя**

Данной системы предназначена для учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». С помощью которой упростится поиск процесс учета заявок на ремонт техники, поиск информации и работа с документами, как для специалистов, так и для других сотрудников больницы.

Автоматизированная информационная система состоит из приложений и СУБД, обеспечивающих создание, наполнение, ведение и модификацию базы данных. Для функционирования АИС на клиентских машинах необходима операционная система Windows 7, на сервере должна быть установлена СУБД MS SQL Server 2012.

Минимальные требования:

Рабочее место - персональный компьютер на базе процессора не ниже Pentium G, с 2 ядрами и частотой не менее 2,2 Ггц, не менее 4048 Мб оперативной памяти, монитор с разрешением не менее 1920 на 1080, манипулятор типа «мышь», клавиатура.

Сервер: Windows Server 2008 R2, Windows Server 2012 или новее. Компьютер с процессором Intel или совместимым процессором с тактовой частотой 2 ГГц и выше (рекомендуется 3 ГГц и выше), не менее 2 ядер. Минимум 4 ГБ ОЗУ (рекомендуется 8 ГБ и выше). 2,2 ГБ свободного места на диске.

Объем базы данных для 100 записей составит примерно 400 Мбайт.

Автоматизированная информационная система реализует такие функции как:

* авторизация пользователей (на уровне СУБД);
* ведение справочников (добавление, удаление, редактирование записей таких справочников как: статус, тип обращения, тип оборудования, тип пользователя, производитель, пользователь);
* ведение учета заявок на ремонт (добавление, удаление, редактирование заявок на ремонт: обращение с параметрами, ремонт, манипуляция);
* учет компьютерного оборудования (добавление, удаление, редактирование данных об оборудования: справочник оборудования, производитель);
* простой и наглядный пользовательский интерфейс;
* формирование отчетов (список заявок, список сотрудников, список оборудования).

Загрузка программы базы данных производится ярлыком «АИС Учет заявок», ссылающимся на исполняемый файл программы.

По истечении некоторого промежутка времени на экране появится главное окно программы с меню пользователя со следующими пунктами: «Справочник», «Пользователь», «Обращение», «Отчёты», «Учет оборудования» и «Выход». Наглядно форма представлена на рисунке 6.



Рисунок 6 – Главное меню

Пункт главного меню «Справочники» содержит подпункты: «Статус обращений», «Тип обращений», «Производители», «Типы пользователей», «Типы оборудования» и «Пользователи» открывающие для работы соответствующие справочники. Работа со всеми справочниками идентична, наглядно форма справочника представлена на рисунках в приложении Н.

В левой части формы находится список всех значений справочника, навигация осуществляется кнопками, расположенными внизу формы. Добавлять значения в справочник можно в правой части формы. Для закрытия окна необходимо нажать на кнопку закрыть.

Пункт главного меню «Пользователь» открывает одноименную форму, позволяющую редактировать список пользователей в системе. Работа с данными аналогична работе со справочником. Наглядно форма «Пользователи» представлена на рисунке 7.

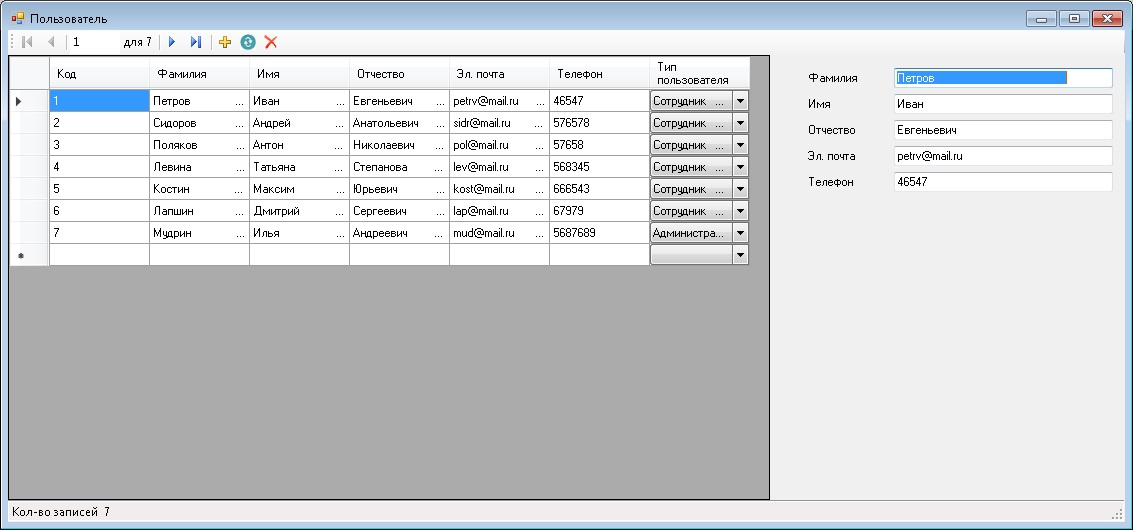


Рисунок 7 – Форма «Пользователи»

Пункт меню «Обращения» открывает одноименную форму «Обращения». Форма предназначена для учета заявок сотрудников на ремонт компьютерного оборудования, а также работы с заявками. Наглядно форма представлена на рисунке 8.

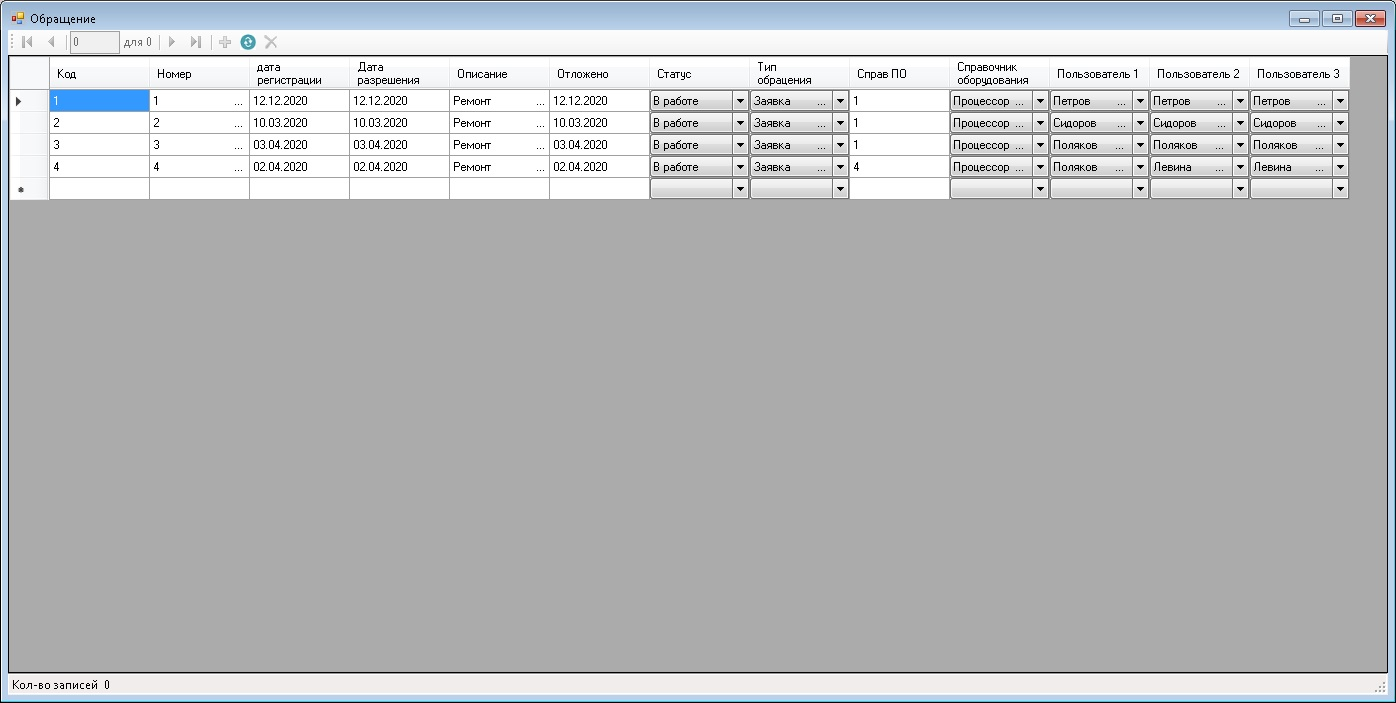


Рисунок 8 – Форма «Обращения»

Пункт главного меню «Учет оборудования» открывает форму «Справочник оборудования» Форма служит для учета оборудования в организации. Наглядно форма представлена на рисунке 9.

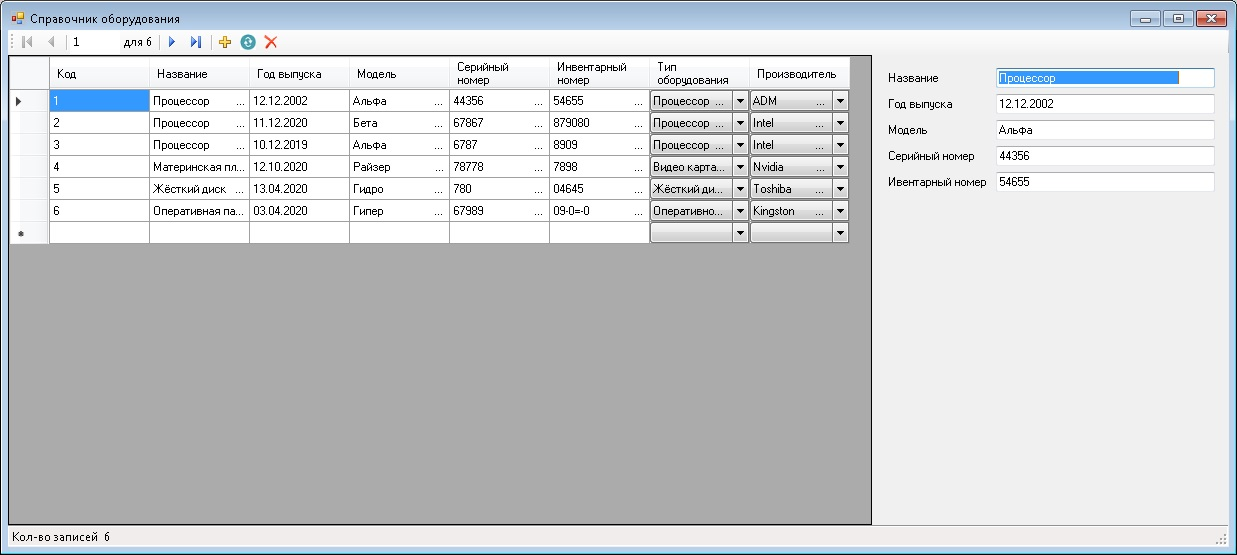


Рисунок 9 – Форма «Справочник оборудования»

В пункте меню «Отчет» содержится список отчетов программы. Наглядно отчеты представлены в приложении В.

В процессе функционирования системы пользователь может столнуться с определенными проблемами. Все имеющиеся сбои выводят на экран разные уведомления. Одинакового текста не бывает. Чтобы проще ориентироваться, разделим существующие ошибки на следующие пункты в соответвии с таблицей 21.

Таблица 21 – Ошибки пользователя

|  |  |  |
| --- | --- | --- |
| Используемые функции | Возникающие проблемы | Решение |
| Сохранение документа или записи | Недостаточно памяти | Увеличить размер выделенной памяти. Для этого запускается адресная строка (Пуск – Выполнить, вводиться фраза cmd). После нажатия «Enter» достаточно ввести фразу bcdedit /set increaseuserva 4096 и подтвердить действие (клавиша «Enter»). Цифра 4096 – новый выделяемый объем «оперативки». Выполняется перезагрузка системы. Проблема должна быть устранена. |
| Отсутствуют права доступа | Ошибка доступа | Для исправления ситуации достаточно изменить роль выбранного профиля в конфигураторе (вкладка «Администрирование – Пользователи»). |
| Сбой при запуске | Отсутствует библиотека Net Framework | Для устранения необходимо установить библиотеку Net Framework 4.5 |

**7 Тестирование и отладка программного комплекса**

Любая программа после разработки должна подвергаться отладке. Отладка – это процесс, направленный на обнаружение и исправление ошибок в программе с использованием процессов выполнения его программ. Процесс включает отладку по каждому модулю в отдельности и в целом. Но ошибки могут появляться и после длительной эксплуатации программы, чтобы их предотвратить, проводят тестирование. Тестирование – процесс выполнения программ программного средства на некотором наборе данных, для которого заранее известен результат или известны правила поведения таких программ.

Для проверки работоспособности программы могут используются следующие методы:

* статистический базируется на правилах структурного построения программ и обработки данных. Проверка выполнения этих правил проводится без изменения объектного кода программы путем формального анализа текста программы на языке программирования. Операторы и команды анализируются в символьном виде, п.э. этот метод еще называют символьным тестированиям.
* детерминированное тестирование, которое требует многократного выполнения программы на ЭВМ с использованием определенных, специальным образом подобранных текстовых наборов данных. При детерминированном тестировании контролируется каждая комбинация исходных данных и соответствующие результаты, а также каждое утверждение в спецификации тестируемой программы.
* стохастическое тестирование используется, когда невозможно перебрать все комбинации исходных данных и проконтролировать результаты функционирования на каждой из них. Стохастическое тестирование предполагает использование в качестве исходных данных множества случайных величин. Стохастическое тестирование применяется для обнаружения ошибок, а для диагностики и локализации применяют детерминированное тестирование с использованием конкретным значением исходных данных из области изменения ранее использовавшихся случайных величин. Стохастическое тестирование наилучших образом подвергается автоматизации путем использования датчиков случайных значений.

При проверке работоспособности всей программы целесообразно использовать стохастическое тестирование, учитывая недостаток времени и в силу этого невозможность проверки всевозможных комбинаций данных. При тестировании программы контролю подверглись следующие режимы:

* ввод данных;
* редактирование данных;
* формирование и вывод отчетов.

Место проведения испытаний – кабинет отдела компьютерного обеспечения. Сначала необходимо заполнить справочную информацию, в которую занести данные, заранее приготовленные заказчиком программы. Это такие данные как: статус, тип обращения, тип оборудования, тип пользователя, производитель, пользователь. Если все данные были занесены, то можно приступать к следующему этапу проверки – получению выходных документов.

Участниками проведения испытаний являются:

– разработчик автоматизированной информационной системы;

– начальник отдела компьютерного обеспечения.

Результаты выполнения программы должны быть известны до начала испытаний. Поэтому все вычисления, которые должны проводиться автоматически, были рассчитаны вручную, а затем сравнивались с полученными результатами работы программы. Заполняется тестовая заявка на ремонт. После этого изменяется статус заявки, формируется выходной документ: список заявок

В процессе испытаний ожидаемый результат совпал с машинным. Следовательно, программы работает правильно и имеет практическое применение.

Целью испытаний является проверка работоспособности, данной АИС. Проверка правильности заполнения списков, формирования отчётов, проверка работоспособности поиска.

**8 Техника безопасности и пожарная безопасность**

Разработка автоматизированной информационной системы учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». Кроме того, компьютеры могут в последующем использоваться в качестве аппаратной базы для работы системы. Ввиду этих причин указанный кабинет, далее будет рассматриваться как рабочее место.

В кабинете находятся 6 персональных компьютеров, объединенных локальной вычислительной сетью являющейся частью сети организации. Компьютеры форм фактора mini-tower расположены на компьютерных столах вдоль трех стен кабинета.

К особенностям освещения кабинета следует отнести наличие окон в двух стенах. Окна, находящиеся в верхней (на плане) стене, выходят на улицу и оборудованы вертикальными жалюзи для регулирования степени освещения аудитории. Окно, находящееся в левой стене, выходит в кабинет администраторов, также освещенный из окон, выходящих наружу здания, создавая дополнительное освещение аудитории, кроме того, облегчая администраторам визуальный контроль за работающими студентами.

Кроме естественного освещения в аудитории применяется искусственное освещение с использованием ламп дневного света, расположенных в два ряда на потолке вдоль более длинной стороны комнаты.

Кабинет объединен коротким коридором с симметричной кабинетом 15 и комнатой отдела кадров, расположенной между ними. Коридор соединяется с основным коридором железной двухстворчатой дверью, открывающейся наружу кабинетом. Кроме того, каждый из кабинетов 14 и 15 имеет одностворчатую дверь, также открывающуюся наружу.

Помещение оборудовано кондиционером, имеющим систему автоматического поддержания заданных климатических условий в помещении, что позволяет поддерживать постоянный уровень температуры и влажности в помещении.

В кабинете есть порошковый огнетушитель модели ОП-5. Рассмотрим более подробно рабочее место пользователя персонального компьютера. Работа с ЭВМ неразрывно связана с рядом факторов, которые могут отрицательно повлиять на здоровье и трудоспособность или представляющими потенциальную опасность. Далее рассмотрим эти факторы, в общем, и применительно к исследуемому кабинету.

Ввиду того, что оператор работает с персональным компьютером, всегда существует опасность поражения электрическим током. Вред здоровью человека причиняется при протекании электрического тока через его тело, что легко может привести к летальному исходу. Увеличивает опасность этого фактора тот факт, что объект, находящийся под напряжением (например, корпус компьютера при возникновении ряда неисправностей и плохом заземлении) невозможно визуально идентифицировать как источник опасности, поскольку никаких внешних изменений не происходит (компьютер в ряде случаев может даже исправно функционировать). Для предотвращения поражений электрическим током следует принять ряд превентивных мер.

Далее рассмотрим числовые характеристики рассматриваемого кабинета в сравнении с их нормативным значением. Сравнение позволит сделать более обоснованный вывод об условиях труда на рабочем месте.

По результатам замеров в кабинете 14 были получены данные, приведенные в таблице 22.

Таблица 22 — Показатели условий труда в помещении

|  |  |  |
| --- | --- | --- |
| Показатель | Значение | Нормативное значение |
| Размеры помещения (длина/ширина/высота), м | 7/5/4 | – |
| Площадь помещения, м2 | 35 | – |
| Количество ПК, шт. | 10 | – |
| Расстояние между мониторами ПК, м | 1–1,5 | 1,2 |
| Площадь на одно рабочее место, м2 | 3,5 | 6 |
| Объем на одно рабочее место, м3 | 14 | 24 |
| Температура в помещениях, С° | 18–22 | 22–24 |
| Оборудование поддержки климатических условий | кондиционер | – |
| Уровень шума, дБ | до 65 | до 50 |
| Оборудование оконных проемов | вертикальные жалюзи | шторы, жалюзи |
| Ориентация световых поемов по сторонам света | восток, северо-восток | север, северо-восток |
| Вид искусственного освещения | прямой косвенный | – |
| Освещенность стола на рабочей поверхности, люкс | 500 | 300–500 |
| Скорость движения воздуха, м/с | 0,2 | 0,1 |
| Относительная влажность воздуха, % | 50 |  |
| Коэффициент отражения света от потолка | 0,7–0,8 | 0,7–0,8 |
| Коэффициент отражения света от стен | 0,5–0,6 | 0,5–0,6 |
| Коэффициент отражения света от пола | 0,3–0,5 | 0,3–0,5 |
| Запыленность, мг/м3 | 3 | 4 |
| Модель огнетушителя | ОП-5 | ОП или ОУ |

Рассматривая полученные данные, можно заметить, что для улучшения условий труда стоит произвести некоторые изменения. Кроме того, следует уменьшить загруженность аудитории, увеличив площадь, приходящуюся на одно рабочее место.

Рассмотрим основные мероприятия по улучшению условий труда в кабинете.

Рекомендуемое расстояние между соседними мониторами должно составлять более 1,2 м. В целях улучшения условий работы стоит увеличить расстояния между персональными компьютерами, находящимися рядом. В первую очередь это касается компьютеров, расположенных у правой стены помещения, поскольку расстояние между ними меньше, чем в среднем по аудитории. Возможным решением этой проблемы будет перенос одной из ЭВМ в другую аудиторию с более подходящими параметрами размещения. В результате этого расстояние между компьютерами увеличиться до 1,4–1,5 м.

Существенное улучшения условий работы можно добиться при замене мониторов на электронно-лучевых трубках жидкокристаллическими мониторами. В последнее время цены на ЖК мониторы существенно снизились, и наладился выпуск «бюджетных» моделей. Таким образом, приобретение такого монитора перестает быть излишней расточительностью, особенно ввиду ряда безусловных преимуществ, предоставляемого ЖК технологией.

Существенным недостатком является расположение кабелей без использования специальных коробов. Таким способом в аудитории расположены кабели питания и коммуникационные сетевые кабели. Отказ от применения коробов может привести к нежелательным последствиям.

Таким образом, применение закрытых коробов для кабелей питания и сетевых кабелей позволит избежать возникновения чрезвычайных ситуаций, потери данных и порчи оборудования.

Во избежание поражений электрическим током необходимо использовать защитное заземление и зануление. Для этого следует использовать систему заземлений в здании, которая, существует в здании. Кроме того, стоит использовать только специальное оборудование электроснабжения, позволяющее обеспечить эффективное использование заземления на каждом приборе, работающим с его использованием. Так стандартная ЭВМ класса IBM PC требует заземления корпуса и использования специальных вилок и розеток питания с заземленным контактом.

Периодически следует проводить профилактический осмотр состояния техники с целью выявления и исправления неполадок в ее работе. Применение этих мероприятий позволит существенно улучшить условия труда в рассматриваемом кабинете, повысить уровень безопасности работы пользователей ЭВМ и снизить влияние неблагоприятных факторов.

**Заключение**

Современные организации производят и накапливают огромные объемы данных. От того, насколько эффективно эта информация используется сотрудниками, руководителями, управляющими органами, зависит качество развития страны в целом и каждого ее территориального субъекта в частности. Поэтому необходимость использования больших, и при этом еще постоянно растущих, объемов информации при решении задач, обуславливает сегодня создание информационных систем в образовательных организациях.

Современные IT широко используют компьютеры, вычислительные сети и всевозможные виды программного обеспечения в процессе управления. Целью внедрения информационных технологий является создание информационных систем (ИС) для анализа и принятия на их основе управленческих решений. Информационные технологии включают два фактора — машинный и человеческий. Конкретным воплощением информационных технологий в основном выступают автоматизированные системы, и лишь в этом случае принято говорить о компьютерных технологиях. Для современных информационных технологий характерны следующие возможности:

* сквозная информационная поддержка на всех этапах прохождения информации на основе интегрированных баз данных, предусматривающих единую унифицированную форму представления, хранения, поиска, отображения, восстановления и защиты данных;
* безбумажный процесс обработки документов;
* возможности совместной работы на основе сетевой технологии, объединенных средствами коммуникации;
* возможности адаптивной перестройки форм и способа представления информации в процессе решения задачи

Результатом работы является автоматизированная информационная система учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум».

В работе был проведён анализ предметной области и обозначены информационные потоки, подлежащие автоматизации. Кроме того, были рассмотрены существующие подходы в решении задачи, определены функции АИС. Выполнен анализ IT-составляющей деятельности организации и определить перспективные направления развития производства процесс учета заявок на ремонт компьютерной техники в организации. В результате комплекса проведённых работ была формализована постановка задачи в форме технического задания.

Кроме того, была разработана модульная архитектура АИС, осуществлена реализация функционального назначения приложения и разработан алгоритм, дан обзор и произведено обоснование выбора программных и технических средств. Были описаны и построена ER-диаграмма, даталогическая модель, произведено физическое проектирование таблиц, было проведено тестирование программного средства и сделан вывод о том, что система работает надежно.

Разработаны руководства программиста и оператора баз данных. Рассмотрены вопросы охраны труда. Рассчитана экономическая эффективность проекта.

Кроме того, в процессе написания техники безопасности и пожарной безопасности были описаны меры безопасности, которыми должен руководствоваться пользователь разработанной подсистемы, прописаны условия для улучшение зрительной работы, снижающие утомляемость и способствующие повышению производительности труда, а также благотворно влияющие на производственную среду, оказывающее положительное психологическое воздействие на инженера по охране труда и техники безопасности, повышающее безопасность труда, качество трудовой деятельности и, как результат, эффективность его работы.

Разработанная АИС представляет совокупность методических и программных средств выполнения следующих функций:

* авторизация пользователей (на уровне СУБД);
* ведение справочников (добавление, удаление, редактирование записей таких справочников как: статус, тип обращения, тип оборудования, тип пользователя, производитель, пользователь);
* ведение учета заявок на ремонт (добавление, удаление, редактирование заявок на ремонт: обращение с параметрами, ремонт, манипуляция);
* учет компьютерного оборудования (добавление, удаление, редактирование данных об оборудования: справочник оборудования, производитель);
* простой и наглядный пользовательский интерфейс;
* формирование отчетов (список заявок, список сотрудников, список оборудования);
* обеспечивает многопользовательский доступ к программному средству.

Применение в эксплуатации данной АИС позволяет увеличить в разы эффективность работы сотрудников отдела компьютерного обеспечения, что обеспечивается автоматизацией разработанного программного модуля ручной работы с документами.

Подсистема позволила:

* намного уменьшить сроки составления документов и отчетов;
* уменьшить количество затрачиваемого времени на ввод и поиск информации;
* оптимизировать хранение справочной информации и автоматизировать доступ к ней;
* снизить количество возможных просрочек выполнения заявок на ремонт компьютерного оборудования;

Использование разработанной конфигурации является экономически выгодным моментом, как для самого разработчика, так и для организации в целом, что описано в разделе экономического обоснования эффективности использования программного продукта.

В данном программном продукте был реализован удобный и простой для пользователя интерфейс. Все действия пользователя при работе с программой сопровождаются интуитивно понятными подсказками, что обеспечивает правильную работу персонала по эксплуатации данной системы.

Благодаря внедрению АИС появится возможность подсчитать количество обращений пользователей в отдел компьютерного обеспечения, снизить частоту поломок техники и гарантировать высокое качество обслуживания техники. Используя полученную информацию, можно спланировать развитие IT инфраструктуры, квалифицировать специалистов и выявить слабые места в организации. Таким образом, АИС позволит заявкам пользователей оперативно поступать в работу, а руководство получает возможность лучше контролировать работу отдела компьютерного сопровождения.

**Список использованных источников**

1. ГОСТ 19.106-78 ЕСПД. Требования к программным документам, выполненным печатным способом. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 16 с.
2. ГОСТ 19.102-77. Стадии разработки ПО. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 12 с.
3. ГОСТ 19.101-77. Виды программных документов. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 18 с.
4. ГОСТ 19.402-78. Описание программы. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 10 с.
5. СанПиН 2.2.2.542-03. Рабочее место программиста.
6. ГОСТ 19.701-90 ЕСПД. Схема алгоритмов, программ, данных и систем. – Введ. Дата введения 1992-01-01. – Москва: Стандартинформ, 2010. – 8 с.
7. ГОСТ 19.301-79. Программа и методика испытаний. – Введ. Дата введения 1981-01-01. – Москва: Стандартинформ, 2010. – 12 с.
8. ГОСТ 19.105-78 ЕСПД. Состав и структура программного документа. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 6 с.
9. ГОСТ 19.503-79. Руководство системному программисту. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 4 с.
10. ГОСТ 19.505-79. Руководство оператору. – Введ. Дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 3 с.
11. ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. – Введ. Дата введения 1992-01-01. – Москва: Стандартинформ, 2010. – 22 с.
12. Сайт государственного бюджетного учреждение здравоохранения "Городская больница" г. Соль-Илецка [Электронный ресурс].: официальный сайт / под ред. А. К. Пензяков. - 3-е изд., перераб. и доп. – Соль-Илецк : Эфис-сервис, 2019. - Режим доступа: [http://crbsol.su/O\_nas/.](http://www.consultant.ru/)- 10.05.2020.
13. Технический блог специалистов ООО"Интерфейс" [Электронный ресурс].: официальный сайт / под ред. А. H. Иванов. - 2-е изд., перераб. и доп. – Москва: ООО «Интерфейс», 2019. - Режим доступа: <https://interface31.ru/tech_it/2019/05/raid-likbez.html>. - 12.05.2020.
14. Обоснование необходимости и проблемы автоматизации. [Электронный ресурс].: блог / под ред. А. H. Иванов. - 2-е изд., перераб. и доп. – Москва: ООО «БоссHR», 2019. - Режим доступа: http://www.bosshr.ru/press-service/publications/news\_detail.php?ID=811. - 12.05.2020.
15. Корецкий, Д.А. Microsoft Corporation. Проектирование и реализация баз данных Microsoft SQL Server 2007. Учебный курс MCAD/MCSE /пер. с англ. – 2-е изд., испр. – М.: Издательско-торговый дом «Русская Редакция », 2018. – 512 с.: ил.
16. Когаловский М. Р. Энциклопедия технологий операционных систем / М.Р. Когаловский. — Москва.: Финансы и статистика, 2018 — 800 с.
17. Зубкова, Т.М. Технология разработки программного обеспечения: учебное пособие / Т. М. Зубкова. - Оренбург: ГОУ ОГУ, 2018. – 101 с.
18. Петров, В. Н. Информационные системы. / В.Н. Петров – СПб.: Питер, 2008. – 688 с.
19. HelpDesk — какую выбрать систему поддержки пользователей [Электронный ресурс]: Электронный блог / под ред. А.И. Миронова- 2-е изд., перераб. и доп. – Москва: ООО «Интертопс», 2019- Режим доступа: <http://blog.sprinthost.ru/2011/07/28/helpdesk-how-to-choose/>
20. Малыхина, М.П. Базы данных: основы, проектирование, использование / М.П. Малыхина. – СПб.: БХВ—Петербург, 2014. – 512 с.
21. Крёнке, Д. М. Теория и практика построения баз данных / Д. М. Крёнке. – СПб.: Питер, 2019. – 800 с.
22. Карпова, Т. С. Базы данных: модели разработка, реализация / Т. С. Карпова. – СПб.: Питер, 2015. – 304 с.
23. Вендеров, А.М. Проектирование программного обеспечения экономических информционных систем: Учебник для вузов / А.М. Вендеров. – М.: Финансы и статистика, 2018. – 352 с.: ил.
24. Арустамова Э.А. Безопасность жизнедеятельности: Учебник / Под ред. проф. Э.А. Арустамова. – 6-е изд., перераб. и доп. – М.: Издательско-торговая корпорация «Дашков и Ко», 2014. – 496 с.
25. Белов С.В. Безопасность жизнедеятельности: Учебник для вузов / С.В. Белов, А.В. Ильницкая, А.Ф. Козьяков и др.; Под общ. ред. С.В. Белова. 4-е изд., испр. и доп. – М.: Высш. Шк., 2014. – 606 с.: ил.
26. Жилин А.Н. Устойчивость функционирования объектов экономики в чрезвычайных ситуациях: Методические указания к практическим занятиям / Жилин А.Н., Денисова Н.Н. – Оренбург: ГОУ ОГУ, 2013. – 47 с.

**Приложение А**

*(обязательное)*

**Расчет экономической эффективности**

**1.1 Расчет себестоимости разработки программного продукта**

С целью определения экономической целесообразности внедрения программного продукта: автоматизированной информационной системы учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум». расчеты, позволяющие определить затраты на разработку программного продукта и рассчитать срок окупаемости предлагаемых изменений. Данные для расчета представлены в таблице А.1.

Таблица А.1 – Исходные данные для расчета себестоимости программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование показателя | Единицы измерения | Обозначение | Значение |
| Норма амортизации компьютера | % | На | 20 |
| Стоимость компьютера | р. | Ск | 32000,00 |
| Стоимость 1 кВт электроэнергии | р. | Скв | 7,01 |
| Мощность компьютера | кВт/ч | Мк | 0,43 |
| Ставка программиста | р. | Стпр | 17800,00 |
| Норма отчислений на дополнительную заработную плату | % | Ндоп | 10 |
| Фонд рабочего времени в месяц | ч | Фвм | 164,2 |

Стоимость машинного часа работы определяется по формуле:

А.1

где См/ч – стоимость машинного часа работы, р.;

Амк – амортизация компьютера за 1 м/ч, р.;

Сэл - стоимость электроэнергии за 1 ч работы, р.;

Амортизация компьютера за 1 м/ч вычисляется по следующей формуле

А.2

где Ск – стоимость компьютера, р.;

На – норма амортизации компьютера, %;

Фв – фонд рабочего времени в год, ч.

Подставив числовые значения, амортизацию компьютера можно рассчитать по формуле (А.2):

Стоимость электроэнергии за 1 час работы вычисляется по следующей формуле

(А.3)

где Сэл - стоимость электроэнергии за 1 час работы, р.;

Мк – мощность компьютера, кВт/ч;

Скв – стоимость 1 кВт электроэнергии, р.

Стоимость электроэнергии рассчитывается по формуле (А.3):

Используя полученные значения, стоимость одного м/ч определяется по формуле (1):

Общая трудоемкость разработки программного продукта составила 144 часа.

(А.4)

Чтобы определить количество человек U, необходимое для выполнения работы, следует воспользоваться формулой:

(А.5)

Подставив все данные в формулу (А.5), получается:

Для решения поставленной задачи в установленные сроки способен справиться один программист.

Далее необходимо рассчитать затраты на оплату труда.

Заработная плата программиста за час определяется по следующей формуле:

, (А.6)

где Стпр – ставка программиста, р.;

Фвм – фонд рабочего времени в месяц, ч.

Для начала рассчитывается заработная плата программиста за час по формуле (А.6):

Дополнительная заработная плата определяется по следующей формуле

, (А.7)

Дополнительную заработную плату рассчитаем по формуле (А.7):

Затраты на оплату труда вычисляются по формуле

(А.8)

где Стр - общая заработная плата, р.;

Зпр - заработная плата программиста, р.;

Здоп - дополнительная заработная плата, р.;

Отч – отчисления на социальные нужды с заработной платы, р.

Вычислим общую заработную плату по формуле (А.8):

Затраты на расходные материалы составляют 3 – 5 % от основной и дополнительной.

Рассчитаем стоимость материалов, покупных полуфабрикатов и изделий:

Накладные расходы составляют 70 % от полной заработной платы.

Накладные расходы составят:

Затраты на текущий ремонт и техобслуживание вычислительной техники составляют 5 – 7 % от заработной платы.

Затраты на текущий ремонт и техобслуживание вычислительной техники составят:

Себестоимость программного продукта рассчитывается по следующей формуле:

Произведем расчет себестоимость разработки программного продукта по формуле (А.9):

**1.2 Обоснование цены программного продукта АИС учета заявок на ремонт компьютерной техники ГАПОУ «Акбулакский политехнический техникум»**

Данный программный продукт является узкоспециализированным, поскольку разрабатывается для нужд одной конкретной организации ГАПОУ «Акбулакский политехнический техникум». Размер желаемой прибыли составляет 30 %.

Рассчитать ожидаемую прибыль можно по следующей формуле

, (А.10)

где Цпр – цена программы, р.;

Пр – ожидаемая прибыль, %.

Рекомендуемый размер ожидаемой прибыли – 30 %.

Следовательно, цена разработанной программы будет рассчитана по следующей формуле (А.10):

**1.3 Определение эффективности от внедрения программного продукта**

Исходные данные для расчета экономической эффективности представим в виде таблицы А.2.

Таблица А.2 - Исходные данные для расчета экономической эффективности программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование показателя | Единицы измерения | Обозначение | Значение |
| Затраты машинного времени на обработку информации | ч | Тм | 4,30 |
| Затраты времени на обработку информации вручную | ч | Тр | 43,20 |

Рассчитаем экономический эффект от внедрения программного продукта.

Стоимость базового варианта обработки администратором рассчитывается по формуле

, (А.11)

Вычислим стоимость базового варианта обработки администратором по формуле (А.11):

Стоимость обработки информации с использованием программного продукта вычисляется по формуле

, (А.12)

Стоимость обработки информации с использованием программы вычислим по формуле (А.12):

Экономический эффект от внедрения программного продукта рассчитывается по формуле

, (А.13)

Экономический эффект от внедрения программного продукта рассчитаем по формуле (А.13):

Срок окупаемости капиталовложений в программный продукт составит

, (А.14)

Определим срок окупаемости капиталовложений по формуле (14):

Коэффициент экономической эффективности от вложенных средств вычисляется по формуле

, (А.15)

Определим коэффициент экономической эффективности от вложенных средств по формуле (А.15):

Показатели экономической эффективности сведем в таблицу А.3.

Таблица 3 – Показатели экономической эффективности разработки программного продукта

|  |  |
| --- | --- |
| Наименование показателя | Значение |
| Себестоимость программы, р. |  |
| Цена программного продукта, р. |  |
| Экономический эффект от внедрения программного продукта, р. |  |
| Срок окупаемости капиталовложений в программный продукт, год. |  |
| Коэффициент экономической эффективности от вложенных средств |  |

**Приложение Б**

*(обязательное)*

**Входные документы**

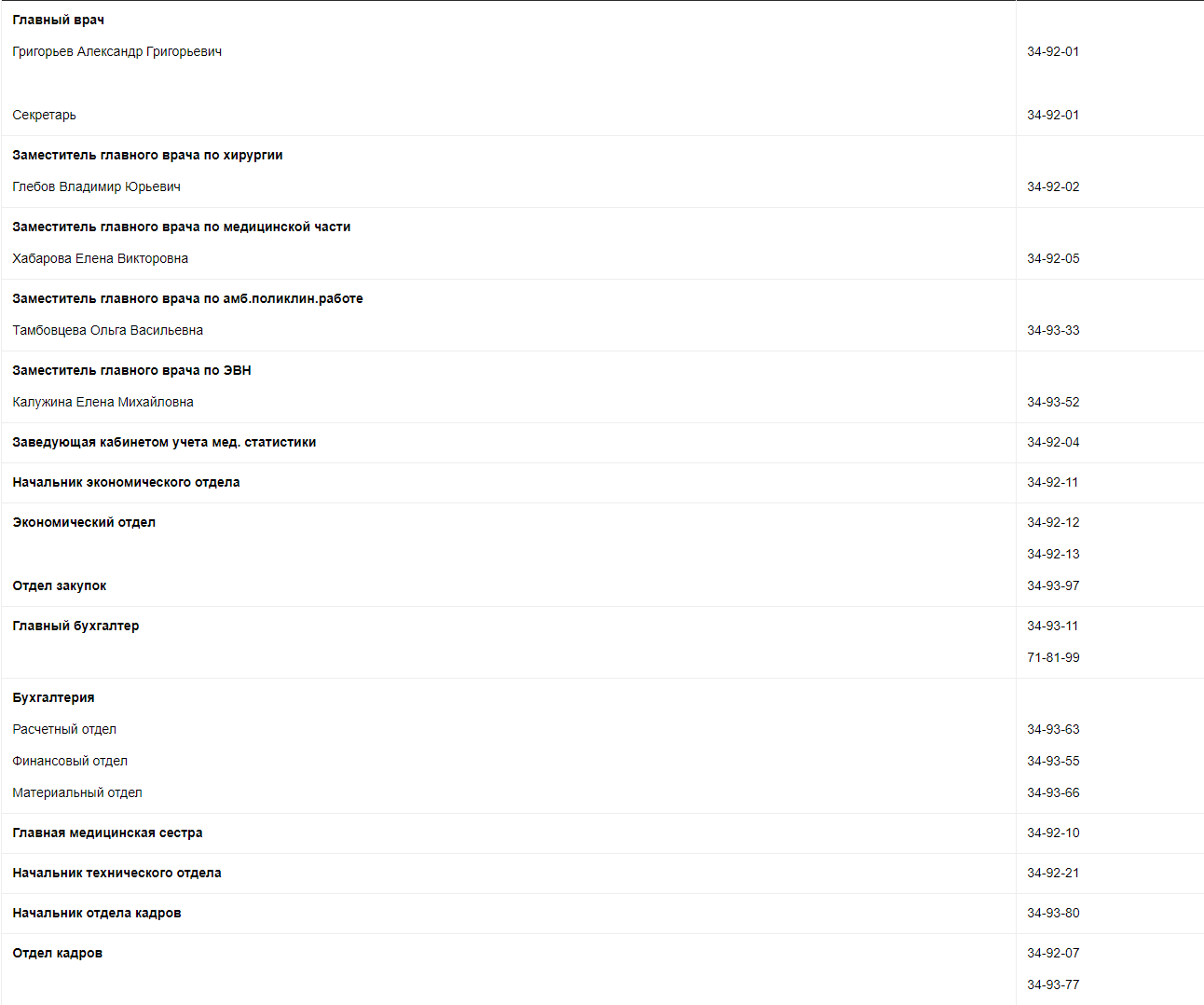


Рисунок Б.1 - Список контактов

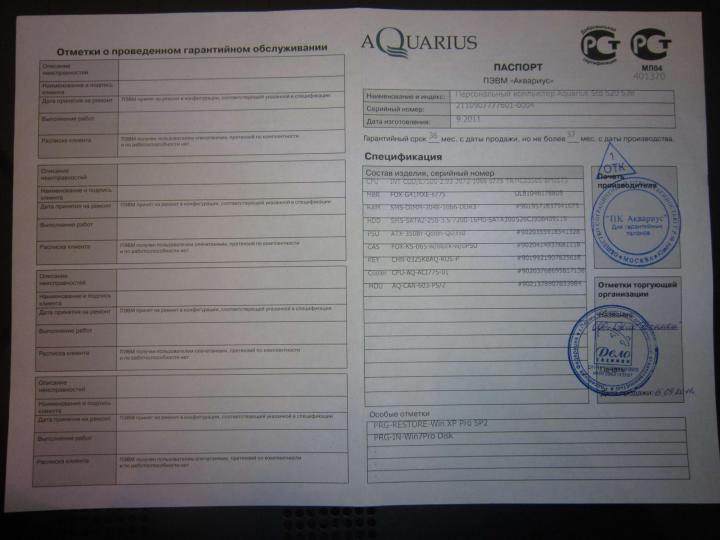


Рисунок Б.2 – Технический паспорт системного блока

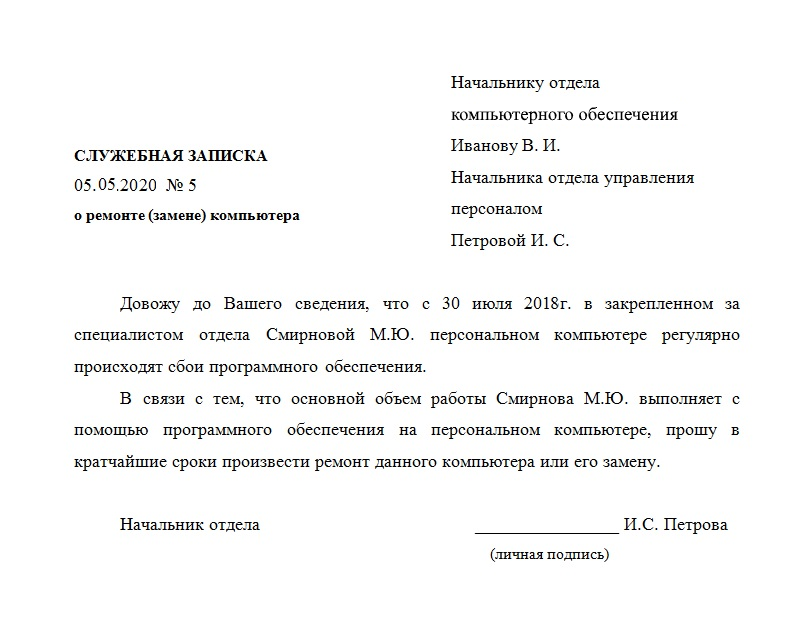


Рисунок Б.3 – Заявка на ремонт компьютера

**Приложение В**

*(обязательное)*

**Выходные документы**

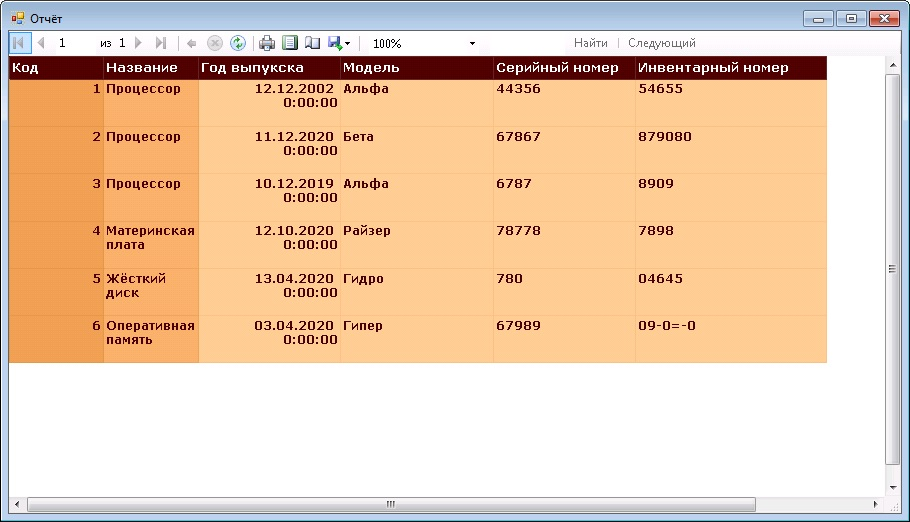


Рисунок В.1 – Список оборудования

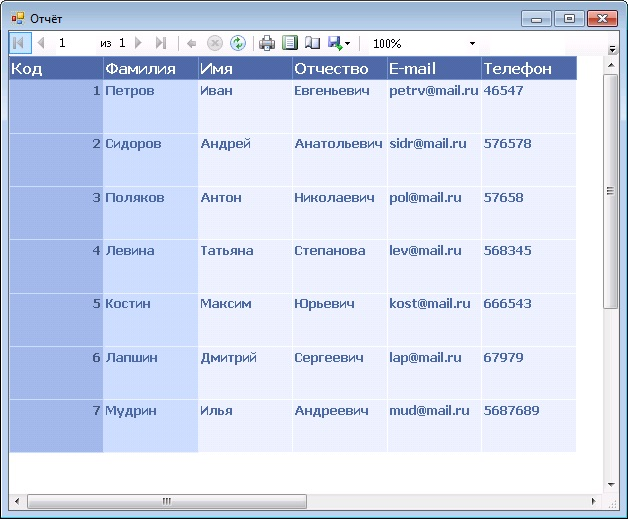


Рисунок В.2 – Список сотрудников

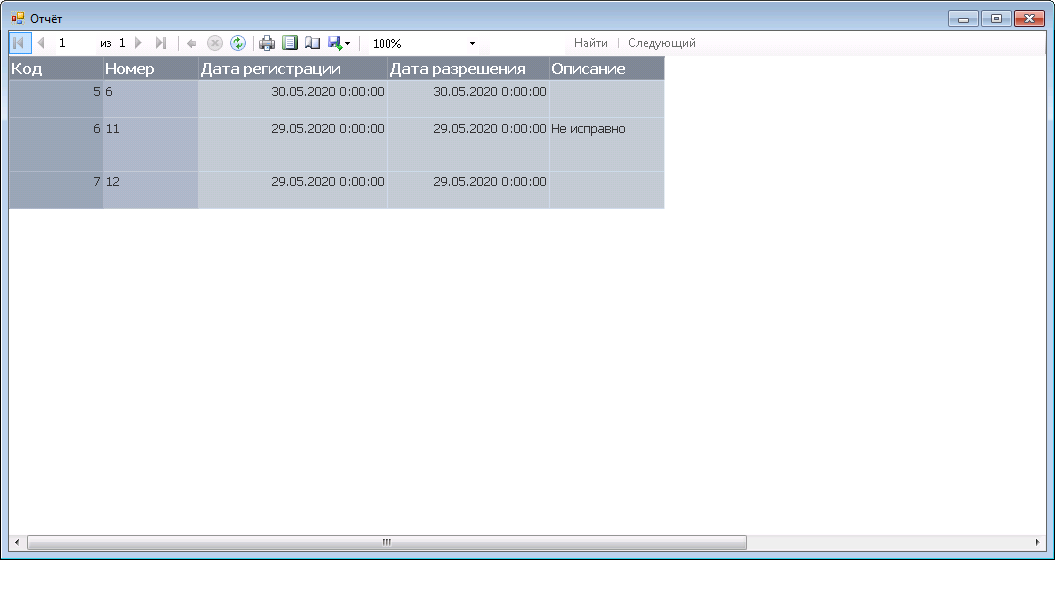


Рисунок В.3 – Отчет по заявкам

**Приложение Г**

*(обязательное)*

Информационная модель

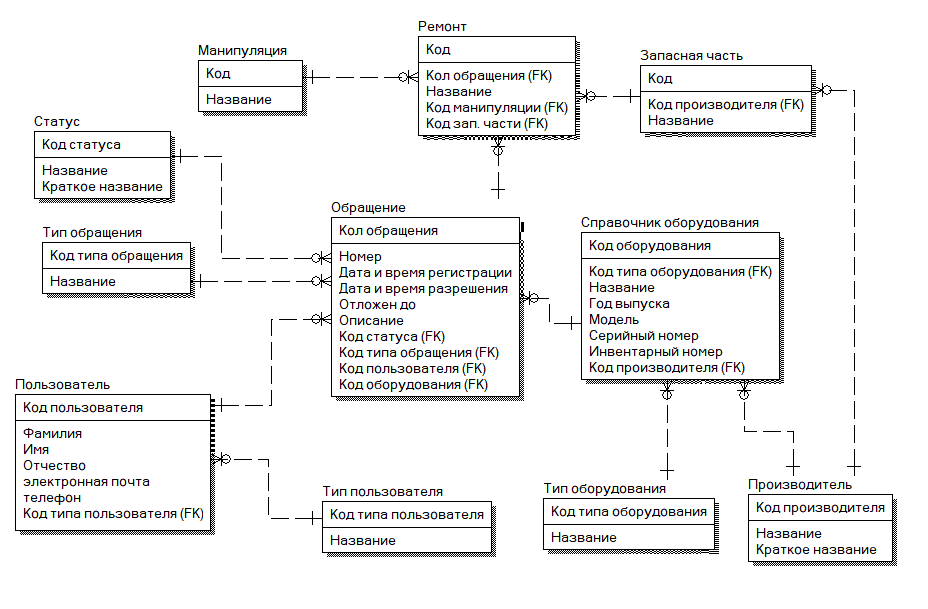


Рисунок Г.1 - Информационная модель АИС

**Приложение Д**

*(обязательное)*

Функциональная модель

начало

helpdesk

helpdesk

Ведение учета заявок на ремонт

текущие действия

ввод команд и данных

номер обращения, дата и время регистрации обращения, дата и время разрешения обращения, описание обращения, критичность, тип обращения, статус, пользователь, ПО, оборудование

helpdesk

ввод команд и данных

helpdesk

Учет компьютерного оборудования

текущие действия

helpdesk

ввод команд и данных

helpdesk

Ведение справочников

текущие действия

helpdesk

ввод команд и данных

helpdesk

формирование отчетов

текущие действия

тип оборудования, производитель, название, год выпуска, модель, серийный номер, инвентарный номер

производитель, пользователь, статус, производитель, тип оборудования, тип пользователя, тип обращения

Условия для формирования отчета

Список заявок

Список сотрудников

Список оборудования

Конец

Рисунок Д.1 – Функциональная модель

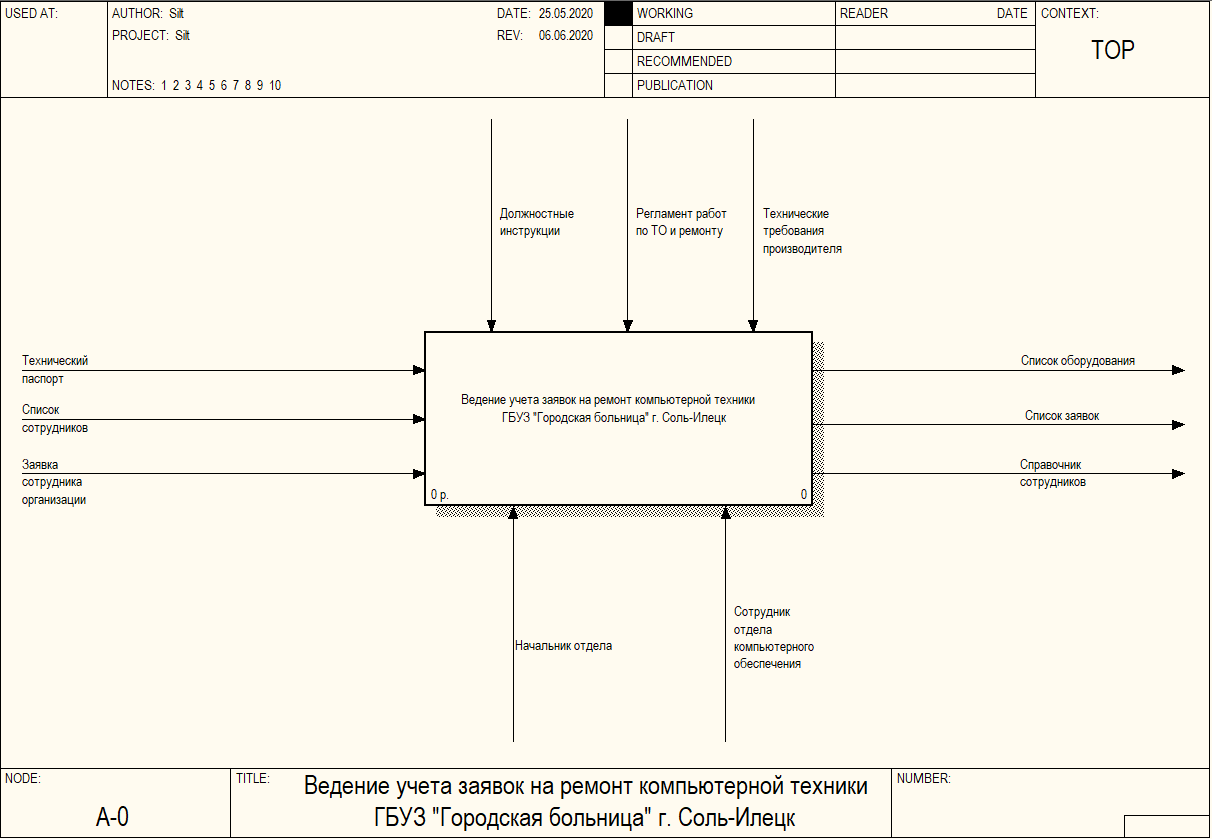


Рисунок Д.2 – Диаграмма А-0

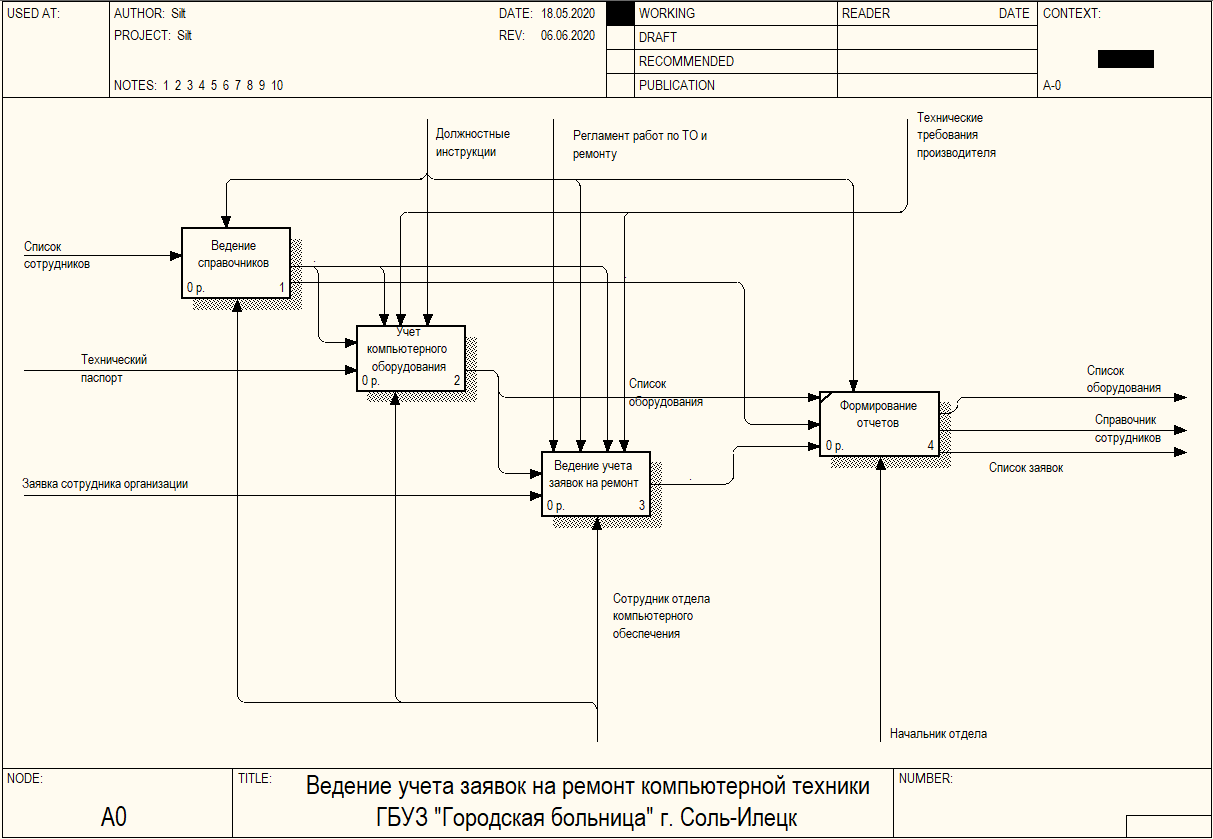


Рисунок Д.3 – Диаграмма А0

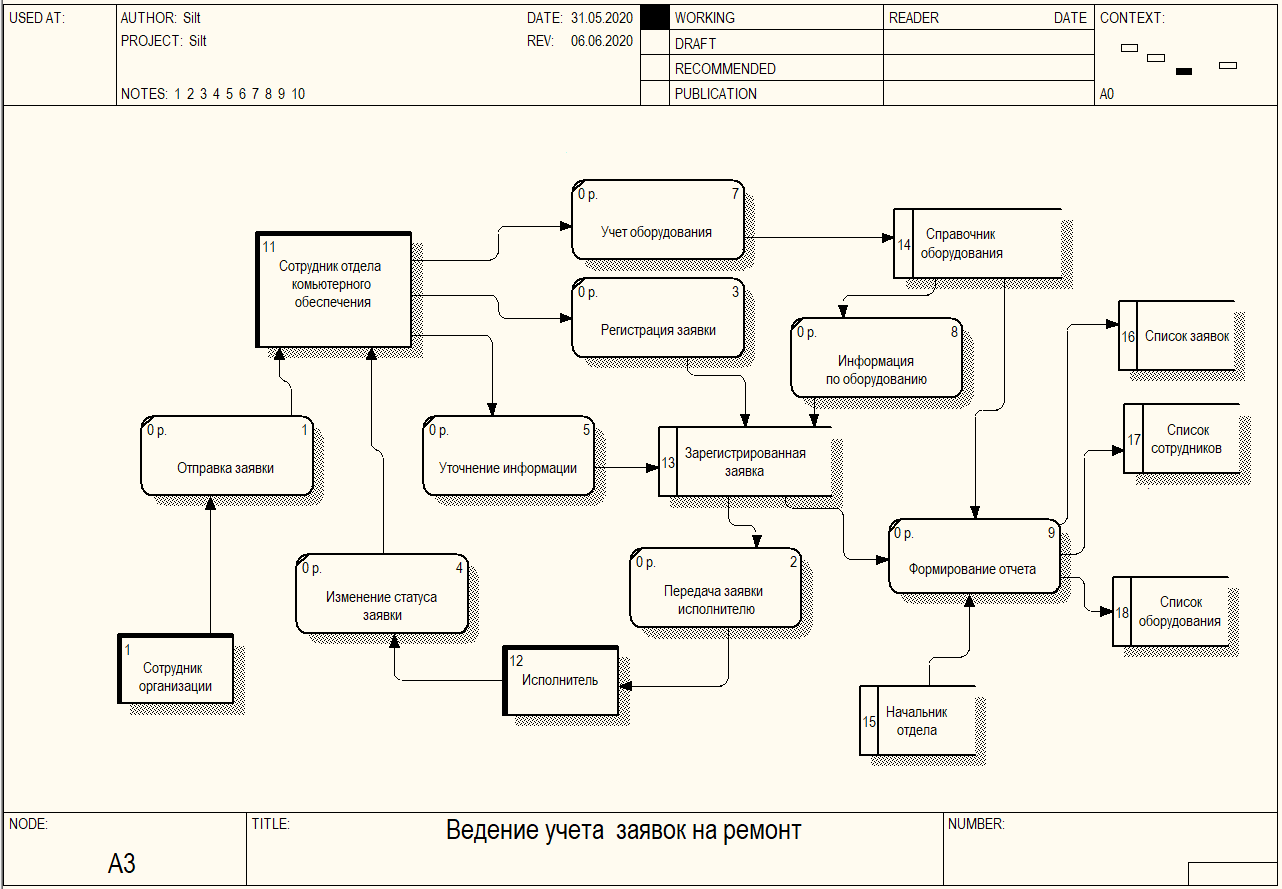


Рисунок Д.4 – Модель потоков данных

**Приложение Е**

***(обязательное)***

# Журнал опытной эксплуатации

|  |  |  |  |
| --- | --- | --- | --- |
| Дата | Используемые функции | Возникающие проблемы | Решение |
| 19.09.20г. | Запуск АИС | Сбой при запуске | Очистить временные файлы и обновить платформу |
| 19.09.20г. | Работа с АИС | Под одним пользователем дает что – то сделать, под другим нет | Настроить права пользователей, очистить кэш |
| 20.09.20г. | Сохранение записи | Недостаточно памяти | На клиентском ПК запустить командную строку от имени администратора, прописать там следующее:  **BCDEdit /set increaseuserva xxxx** – вместо хххх пишите объем виртуального адресного пространства в мегабайтах |

**Приложение Ж**

*(справочное)*

Диаграмма прецедентов

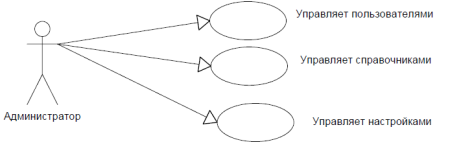


Рисунок Е.1 – Варианты использования АИС для администратора системы

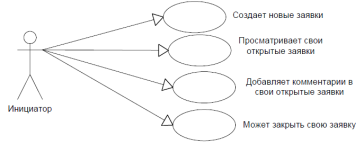


Рисунок Е.2 – Варианты использования АИС для инициатора заявки

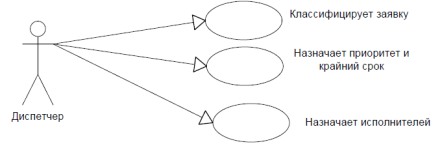


Рисунок Е.3 – Варианты использования АИС для роли диспетчера

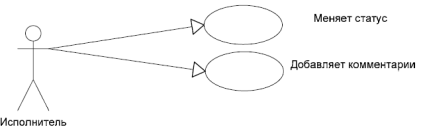


Рисунок Е.4 – Варианты использования АИС для исполнителя

**Приложение Л**

*(справочное)*

Диаграмма деятельности

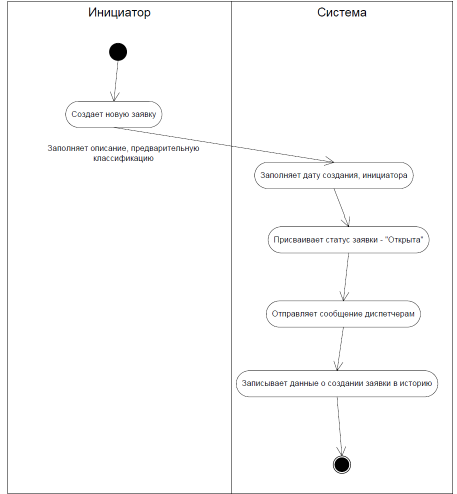


Рисунок К.1 – Создание заявки в системе

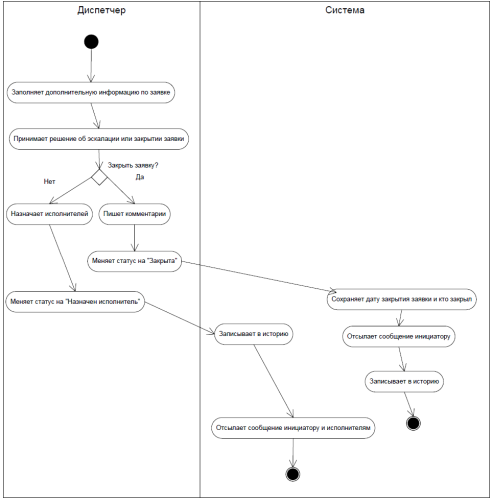


Рисунок К.2 – Обработка заявки диспетчером

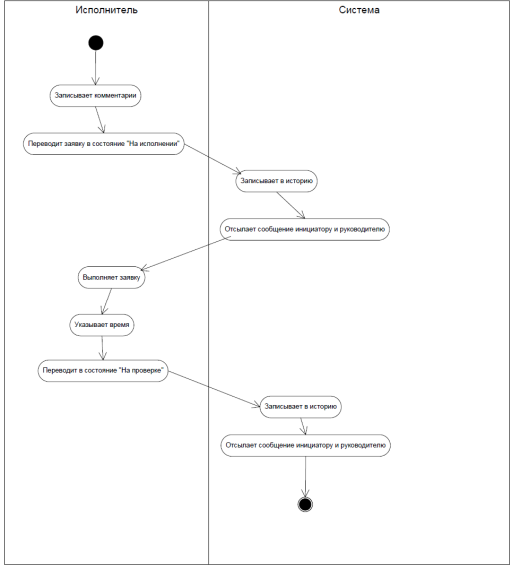


Рисунок К.3 – Обработка заявки исполнителем

**Приложение И**

*(обязательное)*

Листинг 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;

namespace Remont\_PC

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

}

private void статусToolStripMenuItem\_Click(object sender, EventArgs e)

{

StatusForm frm = new StatusForm();

frm.ShowDialog();

}

private void типОбращенияToolStripMenuItem\_Click(object sender, EventArgs e)

{

Type\_obrForm frm = new Type\_obrForm();

frm.ShowDialog();

}

private void манипуляцииToolStripMenuItem\_Click(object sender, EventArgs e)

{

ManuForm frm = new ManuForm();

frm.ShowDialog();

}

private void типПользователяToolStripMenuItem\_Click(object sender, EventArgs e)

{

Type\_polzavtForm frm = new Type\_polzavtForm();

frm.ShowDialog();

}

private void типОборудованияToolStripMenuItem\_Click(object sender, EventArgs e)

{

Type\_oborudForm frm = new Type\_oborudForm();

frm.ShowDialog();

}

private void производительToolStripMenuItem\_Click(object sender, EventArgs e)

{

ProizvodForm frm = new ProizvodForm();

frm.ShowDialog();

}

private void запасныеЧастиToolStripMenuItem\_Click(object sender, EventArgs e)

{

ZapCastForm frm = new ZapCastForm();

frm.ShowDialog();

}

private void выходToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Close();

}

private void справочникОборудованияToolStripMenuItem\_Click(object sender, EventArgs e)

{

SpravOborudForm frm = new SpravOborudForm();

frm.ShowDialog();

}

private void запЧастиToolStripMenuItem\_Click(object sender, EventArgs e)

{

ZapCastForm frm = new ZapCastForm();

frm.ShowDialog();

}

private void ремонтToolStripMenuItem\_Click(object sender, EventArgs e)

{

RemontForm frm = new RemontForm();

frm.ShowDialog();

}

private void обращениеToolStripMenuItem1\_Click(object sender, EventArgs e)

{

ObrashForm frm = new ObrashForm();

frm.ShowDialog();

}

private void пользовательToolStripMenuItem\_Click(object sender, EventArgs e)

{

PolzovForm frm = new PolzovForm();

frm.ShowDialog();

}

private void MainForm\_Load(object sender, EventArgs e)

{

toolStripStatusLabel1.Text = "Время подключения: "+ DateTime.Now.ToString();

}

private void отчёт1ToolStripMenuItem\_Click(object sender, EventArgs e)

{

ReportForm frm = new ReportForm();

frm.ShowDialog();

}

private void отчёт2ToolStripMenuItem\_Click(object sender, EventArgs e)

{

ReportForm2 frm = new ReportForm2();

frm.ShowDialog();

}

}

}

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 Remont\_PC

{

public partial class ObrashForm : Form

{

public ObrashForm()

{

InitializeComponent();

}

private void ObrashForm\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "\_db\_pc\_techDataSet.Polzov". При необходимости она может быть перемещена или удалена.

this.polzovTableAdapter.Fill(this.\_db\_pc\_techDataSet.Polzov);

// TODO: данная строка кода позволяет загрузить данные в таблицу "\_db\_pc\_techDataSet.Ssprav\_oborud". При необходимости она может быть перемещена или удалена.

this.ssprav\_oborudTableAdapter.Fill(this.\_db\_pc\_techDataSet.Ssprav\_oborud);

// TODO: данная строка кода позволяет загрузить данные в таблицу "\_db\_pc\_techDataSet.Type\_obr". При необходимости она может быть перемещена или удалена.

this.type\_obrTableAdapter.Fill(this.\_db\_pc\_techDataSet.Type\_obr);

// TODO: данная строка кода позволяет загрузить данные в таблицу "\_db\_pc\_techDataSet.Status". При необходимости она может быть перемещена или удалена.

this.statusTableAdapter.Fill(this.\_db\_pc\_techDataSet.Status);

// TODO: данная строка кода позволяет загрузить данные в таблицу "\_db\_pc\_techDataSet.Obrash". При необходимости она может быть перемещена или удалена.

this.obrashTableAdapter.Fill(this.\_db\_pc\_techDataSet.Obrash);

toolStripStatusLabel1.Text = "Кол-во записей " +

bindingNavigatorCountItem.Text.Remove(0, bindingNavigatorCountItem.Text.IndexOf(' '));

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

this.obrashTableAdapter.Update(this.\_db\_pc\_techDataSet);

}

}

}

#pragma warning disable 1591

namespace Remont\_PC {

/// <summary>

///Represents a strongly typed in-memory cache of data.

///</summary>

[global::System.Serializable()]

[global::System.ComponentModel.DesignerCategoryAttribute("code")]

[global::System.ComponentModel.ToolboxItem(true)]

[global::System.Xml.Serialization.XmlSchemaProviderAttribute("GetTypedDataSetSchema")]

[global::System.Xml.Serialization.XmlRootAttribute("\_db\_pc\_techDataSet")]

[global::System.ComponentModel.Design.HelpKeywordAttribute("vs.data.DataSet")]

public partial class \_db\_pc\_techDataSet : global::System.Data.DataSet {

private ManipulDataTable tableManipul;

private ObrashDataTable tableObrash;

private PolzovDataTable tablePolzov;

private ProizvDataTable tableProizv;

private RemontDataTable tableRemont;

private Ssprav\_oborudDataTable tableSsprav\_oborud;

private StatusDataTable tableStatus;

private Type\_oborudDataTable tableType\_oborud;

private Type\_obrDataTable tableType\_obr;

private Type\_polzvDataTable tableType\_polzv;

private Zap\_chastDataTable tableZap\_chast;

private global::System.Data.SchemaSerializationMode \_schemaSerializationMode = global::System.Data.SchemaSerializationMode.IncludeSchema;

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

public \_db\_pc\_techDataSet() {

this.BeginInit();

this.InitClass();

global::System.ComponentModel.CollectionChangeEventHandler schemaChangedHandler = new global::System.ComponentModel.CollectionChangeEventHandler(this.SchemaChanged);

base.Tables.CollectionChanged += schemaChangedHandler;

base.Relations.CollectionChanged += schemaChangedHandler;

this.EndInit();

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

protected \_db\_pc\_techDataSet(global::System.Runtime.Serialization.SerializationInfo info, global::System.Runtime.Serialization.StreamingContext context) :

base(info, context, false) {

if ((this.IsBinarySerialized(info, context) == true)) {

this.InitVars(false);

global::System.ComponentModel.CollectionChangeEventHandler schemaChangedHandler1 = new global::System.ComponentModel.CollectionChangeEventHandler(this.SchemaChanged);

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.IncludeSchema)) {

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["Manipul"] != null)) {

base.Tables.Add(new ManipulDataTable(ds.Tables["Manipul"]));

}

if ((ds.Tables["Obrash"] != null)) {

base.Tables.Add(new ObrashDataTable(ds.Tables["Obrash"]));

}

if ((ds.Tables["Polzov"] != null)) {

base.Tables.Add(new PolzovDataTable(ds.Tables["Polzov"]));

}

if ((ds.Tables["Proizv"] != null)) {

base.Tables.Add(new ProizvDataTable(ds.Tables["Proizv"]));

}

if ((ds.Tables["Remont"] != null)) {

base.Tables.Add(new RemontDataTable(ds.Tables["Remont"]));

}

if ((ds.Tables["Ssprav\_oborud"] != null)) {

base.Tables.Add(new Ssprav\_oborudDataTable(ds.Tables["Ssprav\_oborud"]));

}

if ((ds.Tables["Status"] != null)) {

base.Tables.Add(new StatusDataTable(ds.Tables["Status"]));

}

if ((ds.Tables["Type\_oborud"] != null)) {

base.Tables.Add(new Type\_oborudDataTable(ds.Tables["Type\_oborud"]));

}

if ((ds.Tables["Type\_obr"] != null)) {

base.Tables.Add(new Type\_obrDataTable(ds.Tables["Type\_obr"]));

}

if ((ds.Tables["Type\_polzv"] != null)) {

base.Tables.Add(new Type\_polzvDataTable(ds.Tables["Type\_polzv"]));

}

if ((ds.Tables["Zap\_chast"] != null)) {

base.Tables.Add(new Zap\_chastDataTable(ds.Tables["Zap\_chast"]));

}

this.DataSetName = ds.DataSetName;

this.Prefix = ds.Prefix;

this.Namespace = ds.Namespace;

this.Locale = ds.Locale;

this.CaseSensitive = ds.CaseSensitive;

this.EnforceConstraints = ds.EnforceConstraints;

this.Merge(ds, false, global::System.Data.MissingSchemaAction.Add);

this.InitVars();

}

else {

this.ReadXmlSchema(new global::System.Xml.XmlTextReader(new global::System.IO.StringReader(strSchema)));

}

this.GetSerializationData(info, context);

global::System.ComponentModel.CollectionChangeEventHandler schemaChangedHandler = new global::System.ComponentModel.CollectionChangeEventHandler(this.SchemaChanged);

base.Tables.CollectionChanged += schemaChangedHandler;

this.Relations.CollectionChanged += schemaChangedHandler;

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public ManipulDataTable Manipul {

get {

return this.tableManipul;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public ObrashDataTable Obrash {

get {

return this.tableObrash;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public PolzovDataTable Polzov {

get {

return this.tablePolzov;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public ProizvDataTable Proizv {

get {

return this.tableProizv;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public RemontDataTable Remont {

get {

return this.tableRemont;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public Ssprav\_oborudDataTable Ssprav\_oborud {

get {

return this.tableSsprav\_oborud;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public StatusDataTable Status {

get {

return this.tableStatus;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public Type\_oborudDataTable Type\_oborud {

get {

return this.tableType\_oborud;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public Type\_obrDataTable Type\_obr {

get {

return this.tableType\_obr;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public Type\_polzvDataTable Type\_polzv {

get {

return this.tableType\_polzv;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.Browsable(false)]

[global::System.ComponentModel.DesignerSerializationVisibility(global::System.ComponentModel.DesignerSerializationVisibility.Content)]

public Zap\_chastDataTable Zap\_chast {

get {

return this.tableZap\_chast;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.BrowsableAttribute(true)]

[global::System.ComponentModel.DesignerSerializationVisibilityAttribute(global::System.ComponentModel.DesignerSerializationVisibility.Visible)]

public override global::System.Data.SchemaSerializationMode SchemaSerializationMode {

get {

return this.\_schemaSerializationMode;

}

set {

this.\_schemaSerializationMode = value;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.DesignerSerializationVisibilityAttribute(global::System.ComponentModel.DesignerSerializationVisibility.Hidden)]

public new global::System.Data.DataTableCollection Tables {

get {

return base.Tables;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

[global::System.ComponentModel.DesignerSerializationVisibilityAttribute(global::System.ComponentModel.DesignerSerializationVisibility.Hidden)]

public new global::System.Data.DataRelationCollection Relations {

get {

return base.Relations;

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

protected override void InitializeDerivedDataSet() {

this.BeginInit();

this.InitClass();

this.EndInit();

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

public override global::System.Data.DataSet Clone() {

\_db\_pc\_techDataSet cln = ((\_db\_pc\_techDataSet)(base.Clone()));

cln.InitVars();

cln.SchemaSerializationMode = this.SchemaSerializationMode;

return cln;

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

protected override bool ShouldSerializeTables() {

return false;

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

protected override bool ShouldSerializeRelations() {

return false;

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

protected override void ReadXmlSerializable(global::System.Xml.XmlReader reader) {

if ((this.DetermineSchemaSerializationMode(reader) == global::System.Data.SchemaSerializationMode.IncludeSchema)) {

this.Reset();

global::System.Data.DataSet ds = new global::System.Data.DataSet();

ds.ReadXml(reader);

if ((ds.Tables["Manipul"] != null)) {

base.Tables.Add(new ManipulDataTable(ds.Tables["Manipul"]));

}

if ((ds.Tables["Obrash"] != null)) {

base.Tables.Add(new ObrashDataTable(ds.Tables["Obrash"]));

}

if ((ds.Tables["Polzov"] != null)) {

base.Tables.Add(new PolzovDataTable(ds.Tables["Polzov"]));

}

if ((ds.Tables["Proizv"] != null)) {

base.Tables.Add(new ProizvDataTable(ds.Tables["Proizv"]));

}

if ((ds.Tables["Remont"] != null)) {

base.Tables.Add(new RemontDataTable(ds.Tables["Remont"]));

}

if ((ds.Tables["Ssprav\_oborud"] != null)) {

base.Tables.Add(new Ssprav\_oborudDataTable(ds.Tables["Ssprav\_oborud"]));

}

if ((ds.Tables["Status"] != null)) {

base.Tables.Add(new StatusDataTable(ds.Tables["Status"]));

}

if ((ds.Tables["Type\_oborud"] != null)) {

base.Tables.Add(new Type\_oborudDataTable(ds.Tables["Type\_oborud"]));

}

if ((ds.Tables["Type\_obr"] != null)) {

base.Tables.Add(new Type\_obrDataTable(ds.Tables["Type\_obr"]));

}

if ((ds.Tables["Type\_polzv"] != null)) {

base.Tables.Add(new Type\_polzvDataTable(ds.Tables["Type\_polzv"]));

}

if ((ds.Tables["Zap\_chast"] != null)) {

base.Tables.Add(new Zap\_chastDataTable(ds.Tables["Zap\_chast"]));

}

this.DataSetName = ds.DataSetName;

this.Prefix = ds.Prefix;

this.Namespace = ds.Namespace;

this.Locale = ds.Locale;

this.CaseSensitive = ds.CaseSensitive;

this.EnforceConstraints = ds.EnforceConstraints;

this.Merge(ds, false, global::System.Data.MissingSchemaAction.Add);

this.InitVars();

}

else {

this.ReadXml(reader);

this.InitVars();

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

protected override global::System.Xml.Schema.XmlSchema GetSchemaSerializable() {

global::System.IO.MemoryStream stream = new global::System.IO.MemoryStream();

this.WriteXmlSchema(new global::System.Xml.XmlTextWriter(stream, null));

stream.Position = 0;

return global::System.Xml.Schema.XmlSchema.Read(new global::System.Xml.XmlTextReader(stream), null);

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

internal void InitVars() {

this.InitVars(true);

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

internal void InitVars(bool initTable) {

this.tableManipul = ((ManipulDataTable)(base.Tables["Manipul"]));

if ((initTable == true)) {

if ((this.tableManipul != null)) {

this.tableManipul.InitVars();

}

}

this.tableObrash = ((ObrashDataTable)(base.Tables["Obrash"]));

if ((initTable == true)) {

if ((this.tableObrash != null)) {

this.tableObrash.InitVars();

}

}

this.tablePolzov = ((PolzovDataTable)(base.Tables["Polzov"]));

if ((initTable == true)) {

if ((this.tablePolzov != null)) {

this.tablePolzov.InitVars();

}

}

this.tableProizv = ((ProizvDataTable)(base.Tables["Proizv"]));

if ((initTable == true)) {

if ((this.tableProizv != null)) {

this.tableProizv.InitVars();

}

}

this.tableRemont = ((RemontDataTable)(base.Tables["Remont"]));

if ((initTable == true)) {

if ((this.tableRemont != null)) {

this.tableRemont.InitVars();

}

}

this.tableSsprav\_oborud = ((Ssprav\_oborudDataTable)(base.Tables["Ssprav\_oborud"]));

if ((initTable == true)) {

if ((this.tableSsprav\_oborud != null)) {

this.tableSsprav\_oborud.InitVars();

}

}

this.tableStatus = ((StatusDataTable)(base.Tables["Status"]));

if ((initTable == true)) {

if ((this.tableStatus != null)) {

this.tableStatus.InitVars();

}

}

this.tableType\_oborud = ((Type\_oborudDataTable)(base.Tables["Type\_oborud"]));

if ((initTable == true)) {

if ((this.tableType\_oborud != null)) {

this.tableType\_oborud.InitVars();

}

}

this.tableType\_obr = ((Type\_obrDataTable)(base.Tables["Type\_obr"]));

if ((initTable == true)) {

if ((this.tableType\_obr != null)) {

this.tableType\_obr.InitVars();

}

}

this.tableType\_polzv = ((Type\_polzvDataTable)(base.Tables["Type\_polzv"]));

if ((initTable == true)) {

if ((this.tableType\_polzv != null)) {

this.tableType\_polzv.InitVars();

}

}

this.tableZap\_chast = ((Zap\_chastDataTable)(base.Tables["Zap\_chast"]));

if ((initTable == true)) {

if ((this.tableZap\_chast != null)) {

this.tableZap\_chast.InitVars();

}

}

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

private void InitClass() {

this.DataSetName = "\_db\_pc\_techDataSet";

this.Prefix = "";

this.Namespace = "http://tempuri.org/\_db\_pc\_techDataSet.xsd";

this.EnforceConstraints = true;

this.SchemaSerializationMode = global::System.Data.SchemaSerializationMode.IncludeSchema;

this.tableManipul = new ManipulDataTable();

base.Tables.Add(this.tableManipul);

this.tableObrash = new ObrashDataTable();

base.Tables.Add(this.tableObrash);

this.tablePolzov = new PolzovDataTable();

base.Tables.Add(this.tablePolzov);

this.tableProizv = new ProizvDataTable();

base.Tables.Add(this.tableProizv);

this.tableRemont = new RemontDataTable();

base.Tables.Add(this.tableRemont);

this.tableSsprav\_oborud = new Ssprav\_oborudDataTable();

base.Tables.Add(this.tableSsprav\_oborud);

this.tableStatus = new StatusDataTable();

base.Tables.Add(this.tableStatus);

this.tableType\_oborud = new Type\_oborudDataTable();

base.Tables.Add(this.tableType\_oborud);

this.tableType\_obr = new Type\_obrDataTable();

base.Tables.Add(this.tableType\_obr);

this.tableType\_polzv = new Type\_polzvDataTable();

base.Tables.Add(this.tableType\_polzv);

this.tableZap\_chast = new Zap\_chastDataTable();

base.Tables.Add(this.tableZap\_chast);

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

private bool ShouldSerializeManipul() {

return false;

}

**Приложение К**

*(обязательное)*

Контрольный пример

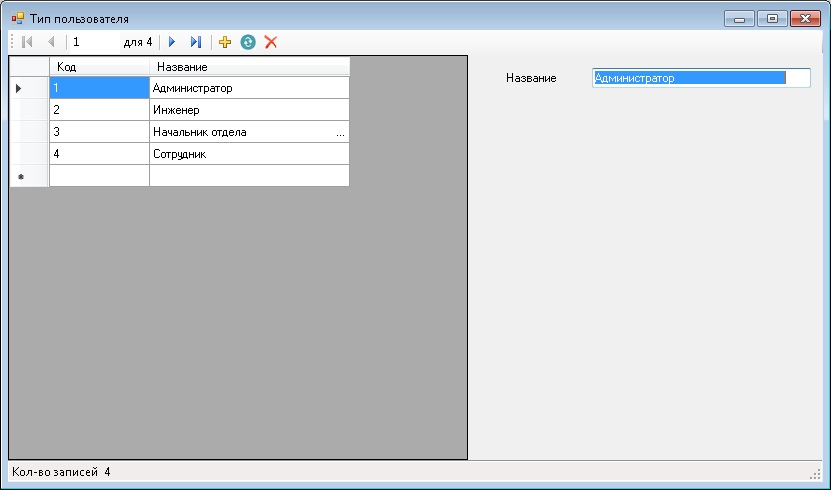


Рисунок М.1 – Форма справочника «Тип пользователя»

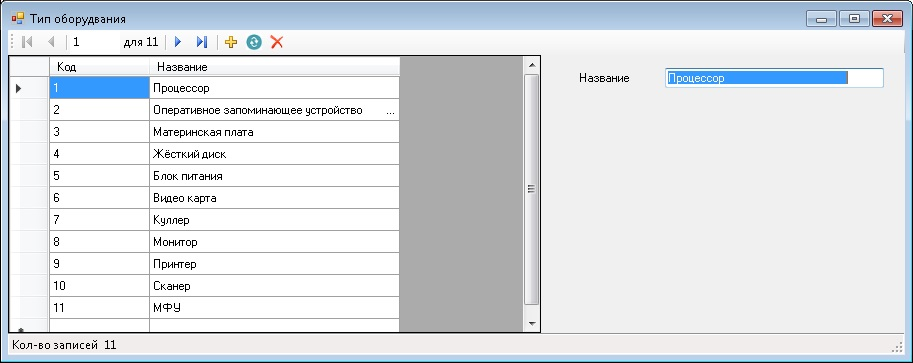


Рисунок М.2 – Форма справочника «Тип оборудования»

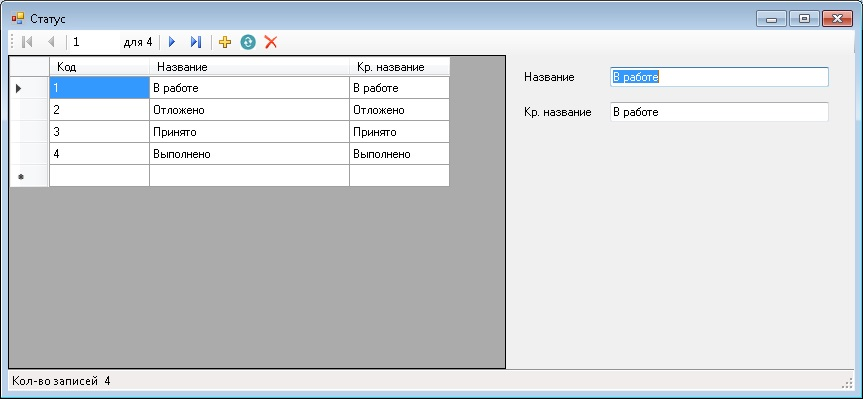


Рисунок М.3 – Форма справочника «Статус»

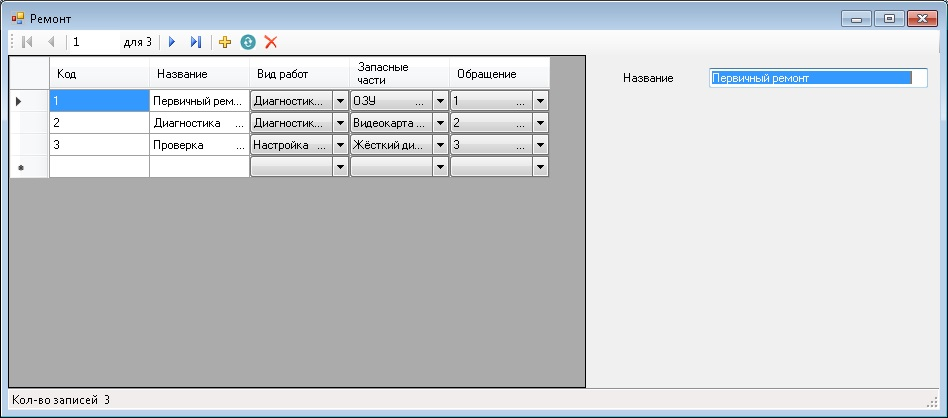


Рисунок М.4 – Форма учета ремонта компьютерного оборудования

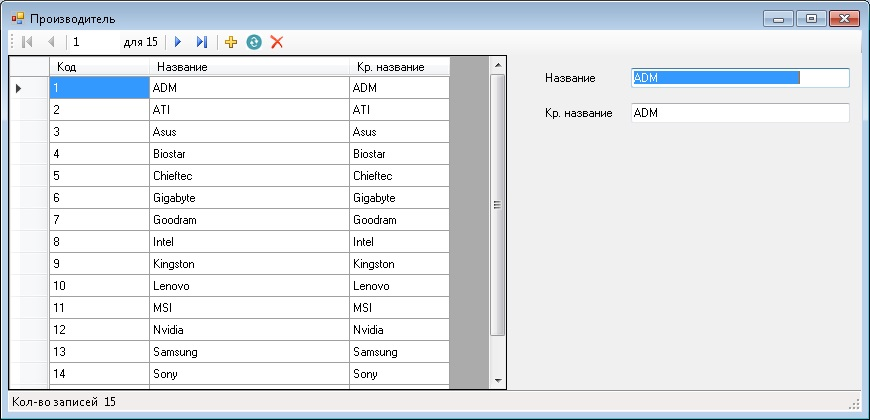


Рисунок М.5 – Форма справочника «Производителей оборудования»

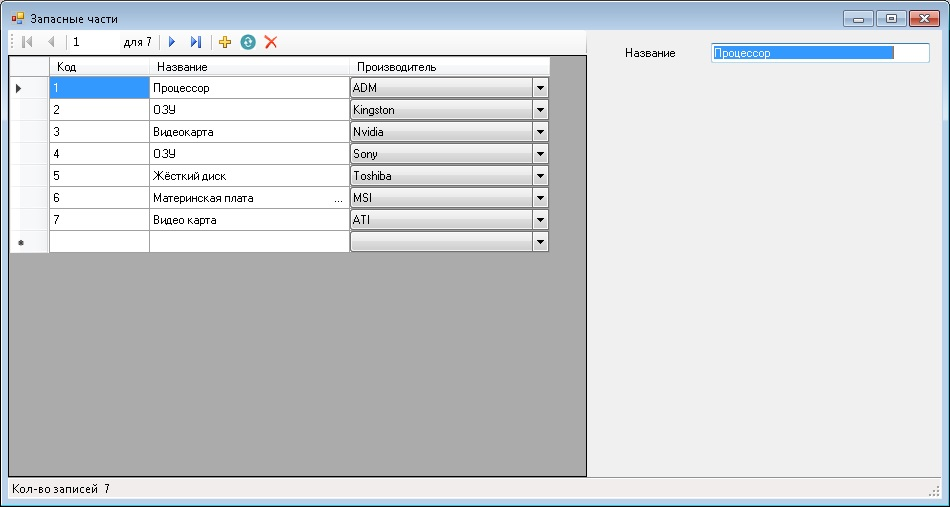


Рисунок М.6 – Форма учета запасных частей

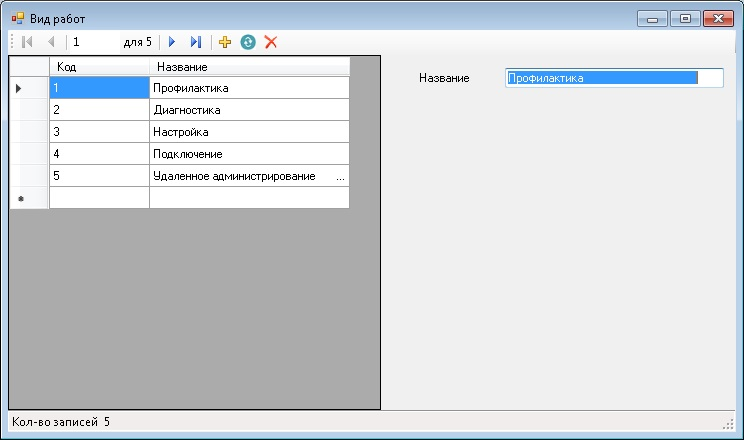


Рисунок М.7 – Форма справочника «Виды работ»

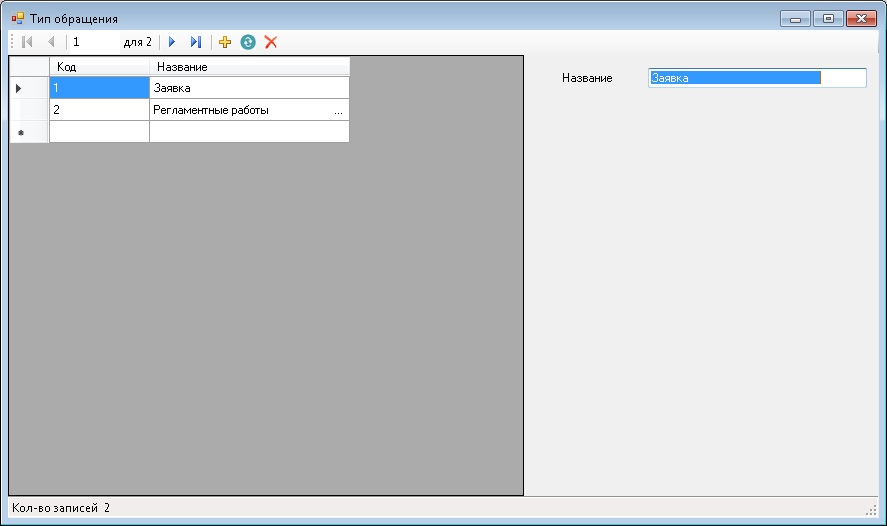


Рисунок М.8 – Форма справочника «Тип обращения»