МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

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

Бизнес-информатика
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка элементов CRM-системы мастерской по ремонту электронной техники (на примере ООО «Электроника Сервис»)

Обучающийся В.А. Дурасов _____
(И.О. Фамилия) (личная подпись)

Руководитель к.п.н., доцент, О.В. Оськина
(ученая степень, звание, И.О. Фамилия)

Тольятти 2022

Аннотация

Выпускная квалификационная работа состоит из 62 страниц, 21 рисунка, 8 таблиц, 23 источников, 1 листа демонстрационного материала.

CRM-система, мастерская, обслуживание заявки, ремонт, статус заявки, электронная техника.

Объектом исследования является – деятельность мастерской по ремонту электронной техники.

Предмет исследования – автоматизация мастерской по ремонту с помощью CRM-системы.

Цель работы – разработать модуль CRM-системы для мастерской по ремонту электронной техники.

Задачи работы:

− исследовать предметную область и выполнить постановку задачи на разработку CRM-системы мастерской по ремонту электронной техники;

− спроектировать систему, разработать структуру базы данных;

− разработать систему, руководство пользователя.

Результатом разработки является модуль для управления взаимоотношениями с клиентами, который поможет сотрудникам мастерской оперативно оповещать клиентов о статусе заявки.

В процессе разработки были использованы: методология канбан, язык программирования C#, среда разработки Visual Studio 2019, база данных MS SQL Server.

В первой главе работы дана технико-экономическая характеристика деятельности мастерской по ремонту электронной техники, рассмотрены основные бизнес–процессы ООО «Электроника Сервис». Проведено моделирование бизнес-процессов в нотации IDEF0 как они осуществляются сейчас («AS–IS»).

Во второй главе проведено инфологическое проектирование, выделены основные объекты системы, на основе чего построены модели данных на уровне концептуальном и логическом представлении данных. Происходит выбор основных инструментов для разработки CRM-системы.

В третьей главе осуществляется физическое проектирование CRM-системы, описывается структура взаимодействия модулей. Также осуществлен выбор методологии разработки и описана архитектура.

Выполнена реализация модуля CRM-системы мастерской по ремонту электронной техники и выполнено тестирование основных функций.

Работа оформлена с помощью текстового редактора Microsoft Office Word 2019 и представлена на флэш-карте (в конверте на обороте обложки). Для оформления презентации был использован Microsoft Office PowerPoint 2019.

Оглавление

Введение

Сервис – комплекс оказываемых услуг определенной компании-производителем товаров. Сервис может быть оказан по гарантии товара либо при повреждении товара или выхода из строя [1].

Современный сервис обязан обеспечить: [2]

– качественно оказанные услуги мастера по потребности потребителя;

– учитывать изменения потребности потребителя, изменения рынка по оказанию услуг.

Для всех этих процессов сервисный центр или любая мастерская, которая занимается ремонтом должны обрабатывать данные, полученные при обслуживании покупателей, вести базу данных, принятых на обслуживание оборудования. Для таких целей создается информационная система, которая обеспечивает сохранность всех этих данных и ускоряет процесс принятия заявок и обслуживание клиентов [1].

Для более эффективного взаимодействия с клиентами придумали CRM-системы, которые помогают персоналу компаний вести общение и контактирование с клиентами, к примеру извещают о статусе заявки.

Объект исследования – деятельность мастерской по ремонту электронной техники.

Предмет исследования – автоматизация мастерской по ремонту с помощью CRM-системы.

Цель выпускной квалификационной работы – разработать модуль CRM-системы для мастерской по ремонту электронной техники.

Задачи выпускной квалификационной работы:

– изучение деятельности мастерской по ремонту;

– проанализировать бизнес–процессы ООО «Электроника Сервис»;

– определить слабый бизнес–процесс, нуждающийся в автоматизации;

– выполнить концептуальное и логическое проектирование системы для решения поставленных задач оптимизации бизнес-процесса;

– разработать и протестировать информационную систему для поддержки деятельности мастерской.

В первой главе работы дана технико-экономическая характеристика деятельности мастерской по ремонту электронной техники, рассмотрены основные бизнес–процессы ООО «Электроника Сервис». Проведено моделирование бизнес-процессов в нотации IDEF0 как они осуществляются сейчас («AS–IS»).

Во второй главе проведено инфологическое проектирование, выделены основные объекты системы, на основе чего построены модели данных на уровне концептуальном и логическом представлении данных. Происходит выбор основных инструментов для разработки CRM-системы.

В третьей главе осуществляется физическое проектирование CRM-системы, описывается структура взаимодействия модулей. Также осуществлен выбор методологии разработки и описана архитектура.

Выполнена реализация модуля CRM-системы мастерской по ремонту электронной техники и выполнено тестирование основных функций.

Глава 1 Функциональное моделирование предметной области

1.1 Характеристика предметной области

Основной функцией мастерской электронной техники является ремонт электронной техники. Чтобы начать ремонт надо сначала принять клиента и само оборудование, которое нуждается в ремонте, выяснить причину поломки оборудования, провести диагностику и оформить заявку на ремонт.

Ремонт может быть произведен по гарантии производителя, если это является гарантийным случаем или без гарантии, в таком случае клиент обязан будет оплатить ремонт.

В заявке обычно указывается название оборудования, модель, дата продажи, также записываются данные клиента для связи. Для корректности заявки, записывается состояние оборудования по предварительному осмотру, заносятся все визуальные сколы и трещины.

Для ведения бизнес-процессов мастерской необходимо провести проектирование и разработку информационной системы для учета заявок и оборудования, а также для оповещения клиентов о статусе заявки.

В списке заявок должны храниться следующие сведения:

– название заявки;

– данные клиента;

– дата заявки;

– данные комплектующих (при необходимости);

– стоимость ремонта;

– данные оборудования.

Наряду со списком заявок в системе должен быть список клиентов, которые подают заявки:

– код клиента;

– ФИО;

– телефон.

Система должна обеспечивать добавление и сохранение данных по заявкам, а также поиск его в списке по номеру или дате принятия.

По каждой заявке должна храниться запись о том, кто из клиентов подал заявку и на какое оборудование.

Так же система должна хранить данные по оборудованию с отметкой, на гарантии оно или нет, список комплектующих на складе, чтобы мастер мог сориентировать клиента по сроку выполнения заявки.

Разработка информационной системы мастерской электронной техники не будет предусматривать ведение каких-либо бухгалтерских расчетов, так как для этого есть довольно удобные специализированные программные продукты такие как 1С: Бухгалтерия.

При разработке информационной системы будет рассматривать только основные процессы мастерской по ремонту электронной техники.

На следующем этапе рассмотрим организационную структуру мастерской по ремонту, взаимодействие между подразделениями.

Мастерская занимается ремонтом электронной техники. Рассмотрим работу небольшой мастерской ООО «Электроника Сервис», которая работает с понедельника по пятницу с 10:00 до 19:00 часов, в субботу и воскресенье – выходной.

Организационная структура мастерской выглядит следующим образом:

– директор;

– бухгалтерия;

– отдел закупок;

– заведующий складом;

– юрист;

– инспектор по кадрам.

На рисунке 1 показана структура управления мастерской по ремонту.

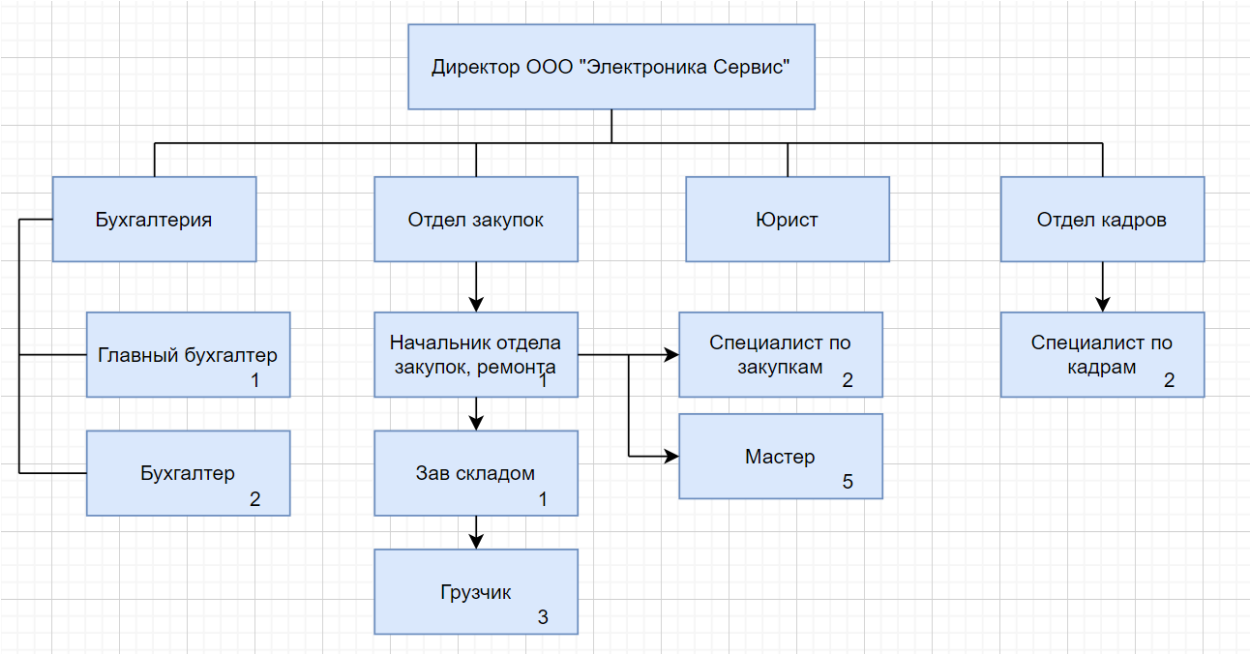

Рисунок 1 – Структура ООО «Электроника Сервис»

1.2 Концептуальное моделирование предметной области

1.2.1 Выбор технологии концептуального моделирования предметной области

Важнейшим этапом разработки автоматизированной информационной системы является моделирование предметной области. К построению модели существуют различные подходы. В настоящее время предпочтение отдается графическим нотациям, так как они позволяют представить моделируемые бизнес-процессы в наиболее наглядной и понятной форме.

Среди самых популярных графических нотаций для моделирования предметной области рассматривались ARIS, IDEF0 и UML.

ARIS (Architecture of Integrated Information Systems) – методология моделирования бизнес-процессов компаний, представленная на рынке одноименным программным продуктом от немецкой компании Software AG.

IDEF0 (Integration Definition for Function Modeling) представляет собой методологию функционального моделирования бизнес-процессов. Диаграммы, разработанные в рамках IDEF0, отображают структуру и функции информационной системы, а также информационные потоки, связывающие эти функции.

UML (Unified Modeling Language) – унифицированный язык моделирования – является языком графического описания, предназначенным для объектно-ориентированного моделирования предметной области, в первую очередь для программных систем. Используется для создания абстрактной модели системы, называемой UML-моделью [3].

Для выбора технологии концептуального проектирования информационной системы было выполнено сравнение указанных нотаций по выбранным критериям. Результаты сравнения представлены в таблице 1.

Таблица 1 – Результаты сравнения технологий моделирования

Критерий	Нотация		
	ARIS	IDEF0	UML
Уровень сложности освоения	высокий	низкий	средний
Простота создания диаграмм	сложно	просто	сложно
Способ хранения диаграмм модели	в объектной базе данных	в нескольких файлах	в одном файле
Подход к проектированию	процессный	функциональный	объектно-ориентированный
Уровни декомпозиции	не ограничены	не ограничены	не ограничены

На основе сравнения для моделирования предметной области и разработки бизнес-модели в существующем варианте «КАК ЕСТЬ» была выбрана нотация IDEF0 как наиболее доступная в изучении и предоставляющая возможность точно и наглядно представить бизнес-процессы ООО «Электроника Сервис».

1.2.2 Моделирование бизнес-процессов предметной области

Клиенты обращаются в мастерскую с заявкой на ремонт электронной техники.

Выделим подсистемы, которые должны функционировать в нашей CRM-системе:

– поступающие данные клиента должны быть зарегистрированы в системе для дальнейшей работы с клиентами;

– при поступлении заявки в системе нужно вносить данные и сохранять данные, для дальнейшего оповещения клиента при готовности товара;

– товары приходящие и отпускающие со склада необходимо фиксировать для учета;

– бухгалтерский учет также необходим для любой организации. Все отчетности бухгалтерии требуется сохранять.

Система мастерской по ремонту будет содержать несколько подсистем (модулей):

Подсистема А1 «Автоматизация регистрации данных о клиентах» предназначена для ввода, хранения и выдачи сведений о клиентах.

Введенные данные клиента используются в дальнейшем для оформления заявки на ремонт электронной техники.

Подсистема А2 «Подсистема регистрации заявки на ремонт» предназначена для отслеживания заявок клиентов для того, чтобы вовремя оповещать клиента о готовности заявки, а также отслеживать статус заявки.

Подсистема А3 «Подсистема автоматизированного складского учета» предназначена для ведения учета комплектующих на складе. В нее входят такие операции как оприходование новых комплектующих, отгрузка комплектующих в сервис, учет комплектующих.

Для бухгалтерии целесообразно будет приобрести готовый модуль, которые предлагают на рынке, они имеют достаточно удобный интерфейс и готовые формы для отчетности, к примеру «1С: Бухгалтерия».

Список пользователей ИС мастерской по ремонту:

– администратор;

– мастер;

– начальник отдела закупок;

– заведующий складом;

– бухгалтерия.

Диаграмма IDEF0, описывающая деятельность мастерской по ремонту электронной техники, показана на рисунке 2.

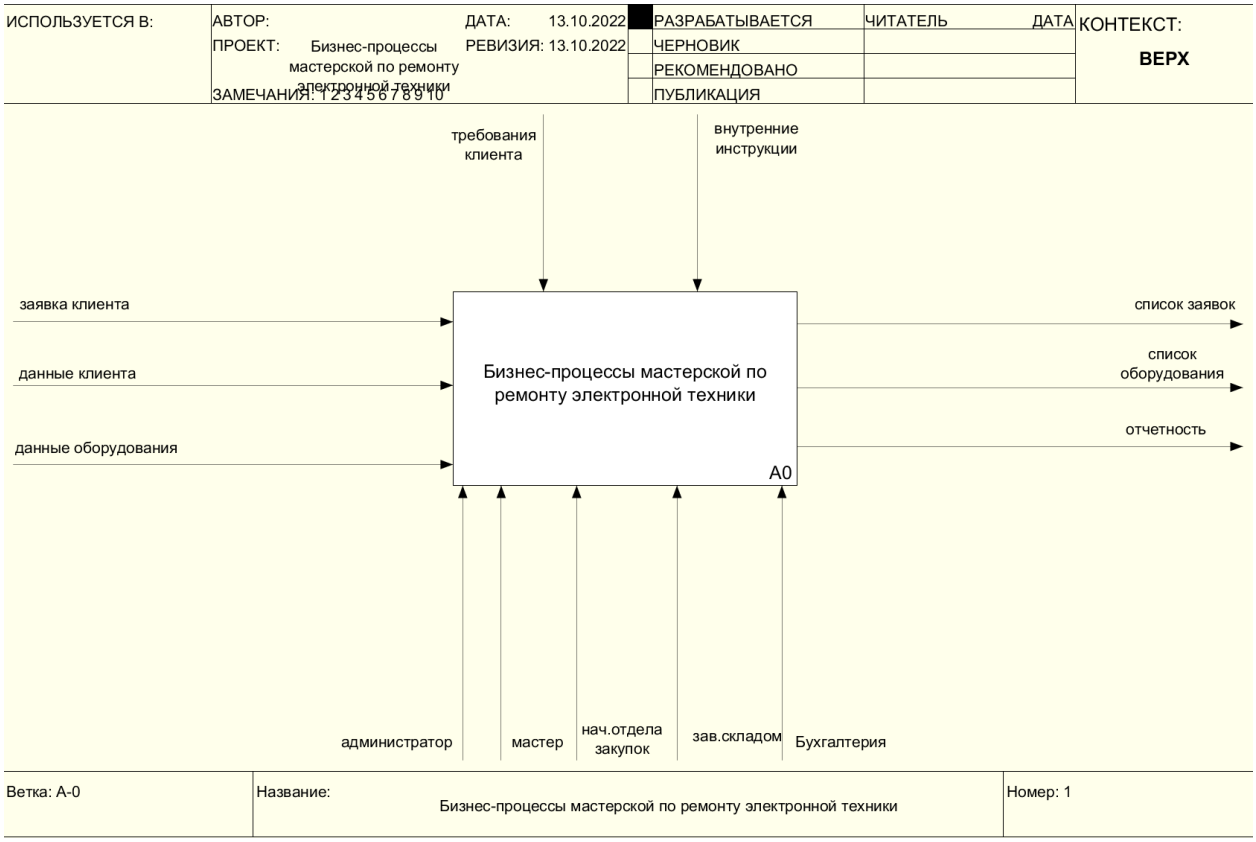

Рисунок 2 – Контекстная модель «КАК ЕСТЬ» бизнес-процессов мастерской по ремонту электронной техники

Чтобы представлять процесс более подробно проведем декомпозицию процесса, разбив его на четыре процесса:

– зарегистрировать данные клиента;

– зарегистрировать заявки клиента;

– внести данные в складской учет;

– внести данные в бухгалтерский учет.

Диаграмма декомпозиции показана на рисунке 3.

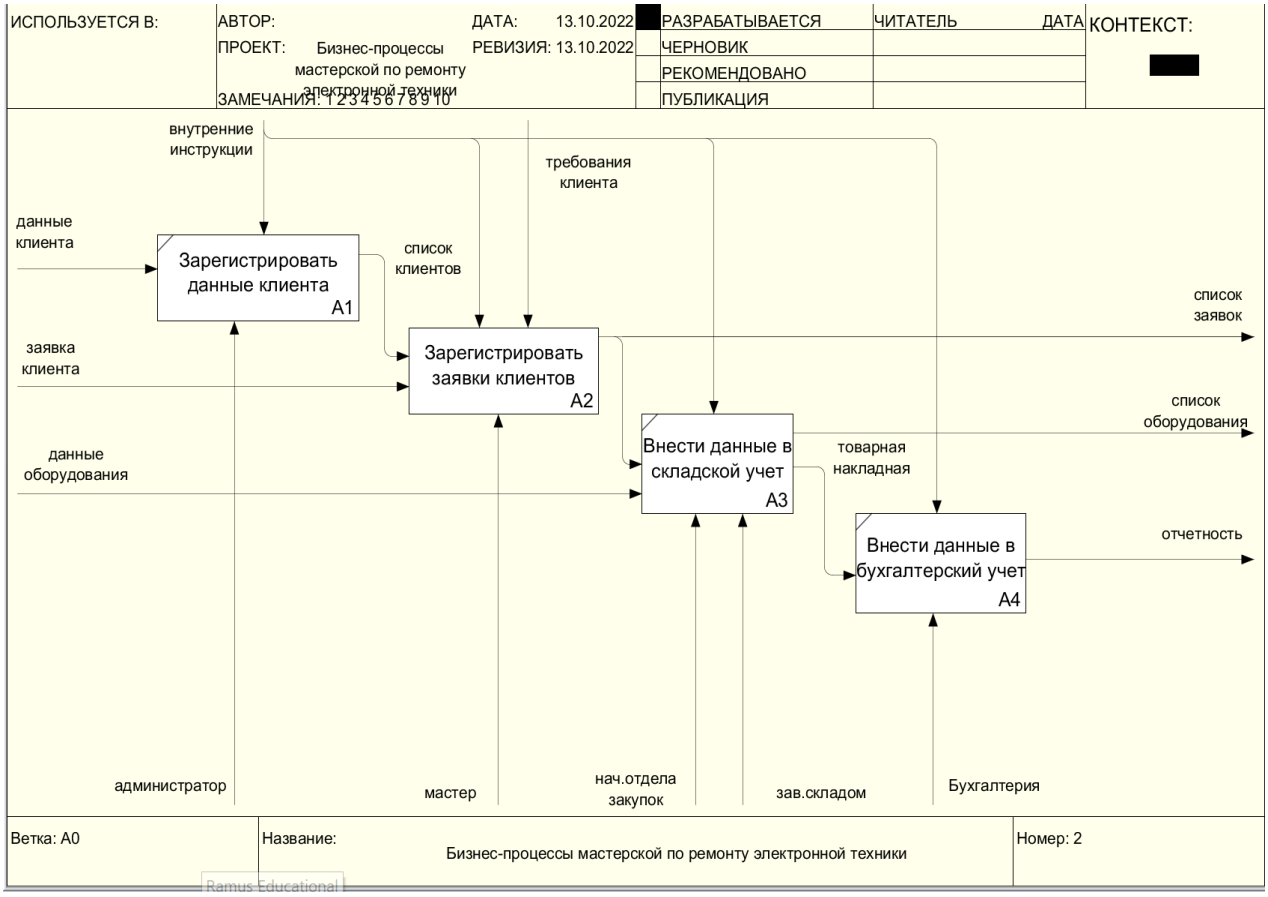

Рисунок 3 – Декомпозиция модели «КАК ЕСТЬ» первого уровня

Представим декомпозицию одной из подсистем «Зарегистрировать заявку клиента», рисунок 4.

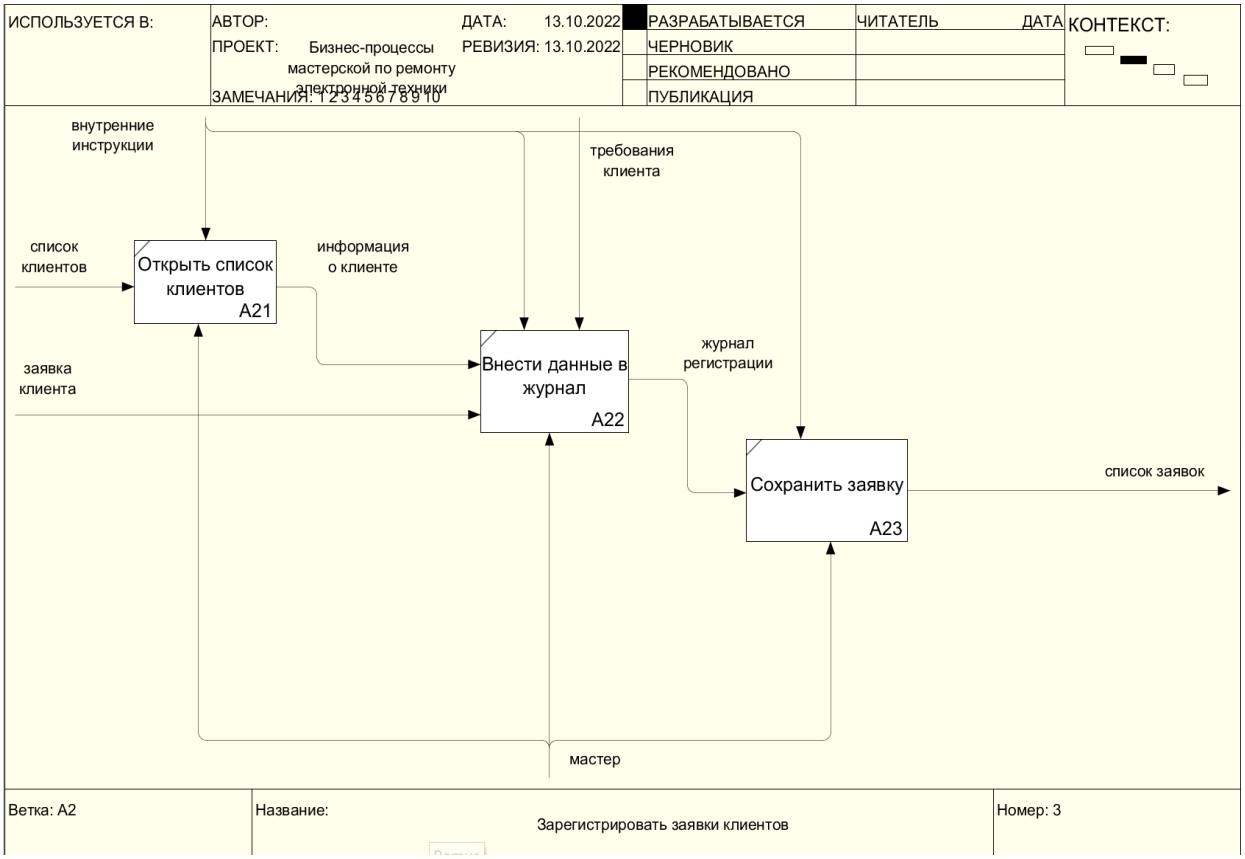

Рисунок 4 – Декомпозиция бизнес-процесса «Зарегистрировать заявку клиента»

На основе диаграмм информационной модели, построенных в нотации IDEF0, можно сделать вывод и определить какие информационные потоки нуждаются в автоматизации:

– ведение данных клиента (ФИО, телефон);

– ведение списка заявок (данные заявки, количество, дата заявки, оборудование, статус);

– ведение списка комплектующих в наличии (номер комплектующего, количество, цена);

– печать сформированной заявки;

– формирование отчетов.

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

Анализ при построении моделей «КАК ЕСТЬ» выявил проблемы мастерской:

– обслуживание одного клиента может затянуться на несколько часов. Заявки обрабатываются медленно за счет хранения информации в разных местах на бумажных носителях, а также нет доступа к данным других отделов;

– выполнение заявок происходит без должного контроля и учета, Распределение комплектующих может расходоваться без должного учета, что может привести к растрате средств [4];

– клиент не имеет четкого понимания, что происходит с его заявкой и в каком она сейчас статусе, от этого клиенты постоянно звонят и узнают, когда будет готова техника.

Для достижения миссии мастерской необходимо разработать информационную систему, которая позволит эффективно работать с клиентами и сократить потерю данных, что улучшит обслуживание клиентов. Исходя из списка приведенных критических факторов успеха, сделаем вывод, что основными критериями разрабатываемой системы должны быть скорость обработки заявок, сокращение ошибок, а также, оптимизация работы сотрудников мастерской. Целями разрабатываемой информационной системы и будет реализация обоих этих критериев.

Задачи информационной системы:

– построить эффективную систему работы с клиентами –финансовая благополучность компании зависит от качества и скорости работы мастерской, оценка эффективности работы мастерской будет производиться по скорости и качеству обслуживания клиентов. Для этого необходимо построить обслуживание таким образом, чтобы сократить время

обслуживания клиента на этапе принятия заявки и ее выполнения с учетом работы всех подразделений одновременно;

– для уменьшения ошибок при обработке информации необходимо провести процесс автоматизации – спроектировать информационную систему, которая будет содержать единую базу данных для хранения всей поступающей информации о клиентах, заявках, комплектующих, возможности реализации групповой работы с информационной системой;

– обеспечить доступ к данным каждому сотруднику, разрешить работу с информационной системой и базой данных.

1.2.4 Бизнес-процессы мастерской

В работе мастерской можно выделить три основных бизнес-процесса: «Принятие заявки», «Заказ комплектующих», «Выдача техники клиенту». В результате первого бизнес-процесса формируется заявка на ремонт электронной техники. В результате второго бизнес-процесса создается заказ на комплектующие на склад. В результате третьего бизнес-процесса клиент и мастерская подписывают акт выполненных работ.

На основе полученной заявки, запускается бизнес-процесс «Принятие заявки», рисунок 5.

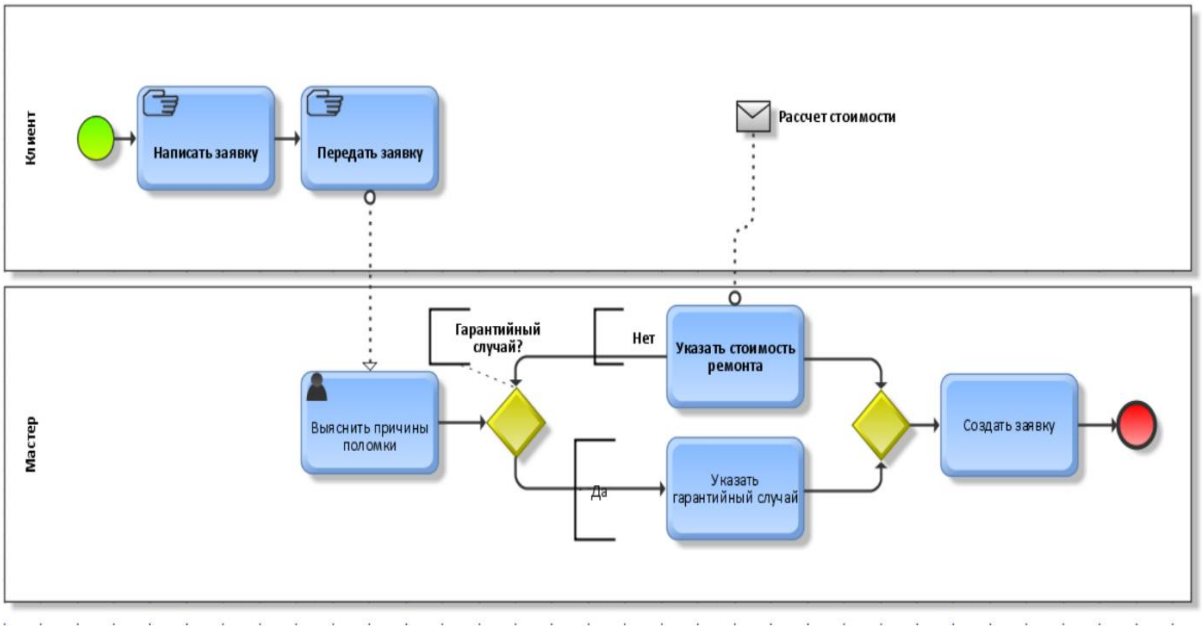

Рисунок 5 — Бизнес-процесс «Принятие заявки»

Если ремонт подразумевает замену комплектующих, то запускается бизнес-процесс «Заказ комплектующих», рисунок 6.

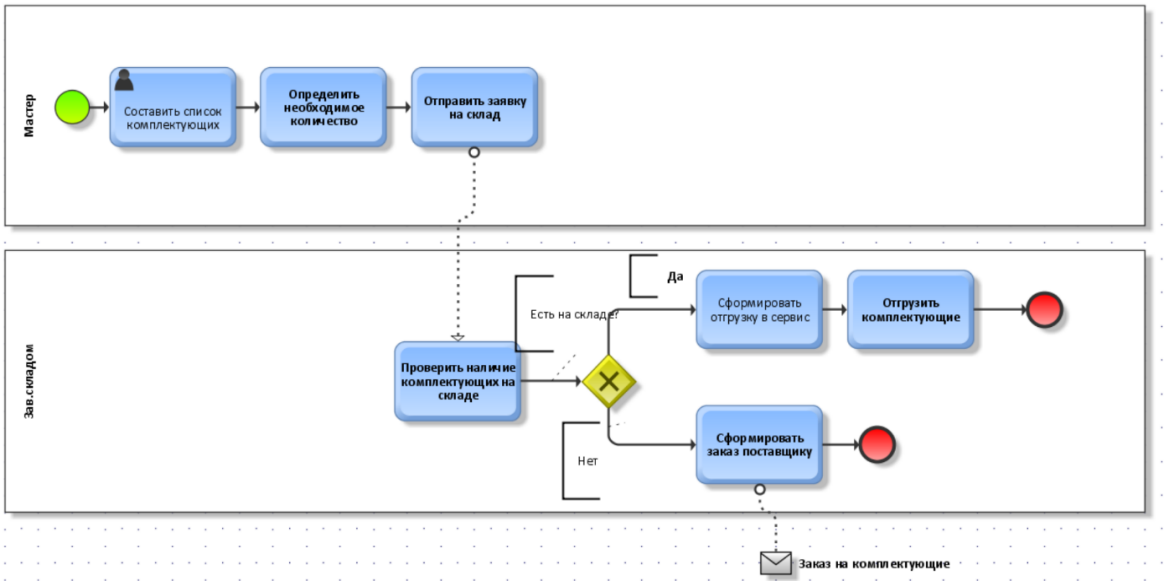

Рисунок 6 — Бизнес-процесс «Заказ комплектующих»

После завершения первых двух бизнес-процессов запускается бизнес-процесс «Выдача техники клиенту», рисунок 7.

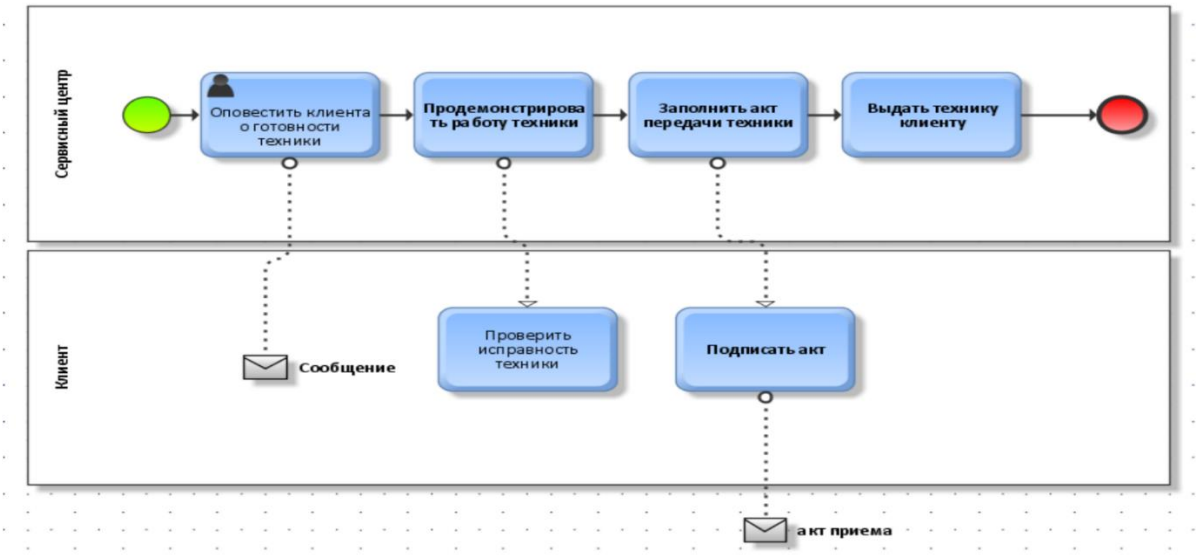

Рисунок 7 — Бизнес-процесс «Выдача техники клиенту»

В целом процесс выдачи техники клиенту выполняется не плохо, остальные бизнес-процессы достаточно медленные и могли бы выполняться согласованнее и быстрее, так как связь между отделами и сотрудниками сервиса не налажена, процессы выполняются медленно. Часто данные приходится передавать по телефону.

Для устранения выявленных недостатков необходимо провести оптимизацию бизнес-процессов. Оптимизация должна проводиться в соответствии с миссией организации, а также поставленными целями и задачами информационной системы.

Для понимания между сотрудниками мастерской и клиентами надо разработать наиболее эффективную систему.

CRM-система должна обеспечивать:

– сбор нужной информации в понятной форме;

– регистрация заявки незамедлительно;

– обновление статуса заявки для отслеживания изменений по заявке;

– контактирование с пользователем по вопросам касающихся заявки;

– доступность для любого пользователя системы [5].

Клиенты должны быть уверены в том, что, сообщив один раз свою мысль, она будет записана, рассмотрена, будет ответ или решение.

Очень важный элемент системы это – база данных, где будут храниться все данные объектов, чтобы отслеживать те действия, которые над ними будут выполняться.

1.3 Анализ существующих разработок

Представленные системы используют различные базовые платформы и технологии, проанализируем достоинства и недостатки некоторых из них.

Облачная платформа управления сервисным обслуживанием «HubEx» от компании Smart Service Solution [6].

Платформа позволяет:

– реагировать на обращения;

– распределять заявку по мастерам и присваивать статус заявки;

– отслеживать историю оборудования и ранние ремонты, если таковые были;

– возможность отвечать на обращения в мобильной версии [7].

Приведены лишь некоторые функции системы HubEx, наиболее приближенные к требованиям, система многофункциональная и предполагает первоначальное обучение для дальнейшего пользования.

Программный продукт Naumen Service Desk от компании Naumen. Предназначен для автоматизации процессов управления ИТ и сервисным обслуживанием на предприятиях [8]. Позволяет автоматизировать работу службы Service Desk и реализовать на практике множество процессов, таких как:

– отслеживание и регистрация всех заявок и обращений;

– быстрое управление обращениями и заявка, ориентация между сотрудниками мастерской.

Продукт очень большой и многофункциональный, для его использования компании необходимо отправлять своих сотрудников для обучения использования системы.

Облачная система Кларис.

Возможности системы:

– единая точка обращения за помощью к специалистам;

– интуитивный и удобный для пользователей интерфейс даёт возможность делать запросы в службу поддержки быстро и легко;

– регистрация заявок по e-mail;

– круглосуточный доступ к системе;

– доступ к данным ограничивается ролью пользователей в системе, каждый из которых видит только то, что необходимо для их работы;

– настройка соглашения об уровне предоставления сервисной услуги;

– хранение приложений [9].

Разработка очень интересная, но дорогостоящая, разовое подключение на 5 пользователей составляет 35000 рублей, за каждого нового пользователя нужно доплачивать еще 7000 рублей, так же идет ежегодное сопровождение, которое ведет еще ряд расходов.

На рынке существует множество интересных разработок по обслуживанию заявок, но все они требуют больших вложений денежных средств и обучения сотрудников для использования системы, что крайне нежелательно.

Проанализировав некоторые аналогичные разработки отечественных компаний разработчиков, приведем данные в таблице, где покажем основные функции необходимые и важные для разрабатываемого приложения.

В таблице 2 проанализируем такие функции как: доступный интерфейс, простота использования, сохранение данных и доступность в цене.

Результаты сравнения функциональных возможностей систем приведены в таблице. При этом используются следующие значения:

– «0» – возможность не реализована;

– «0,5» – реализация с помощью стороннего ПО, дополнительного ПО разработчика или неполная реализация возможности;

– «1» – возможность реализована.

Таблица 2 – Результаты сравнения программ

Основные отличия	«HubEx»	Naumen Service Desk	Кларис
1	2	3	4
Доступность в цене	0,5	0,5	0,5
Дружественный интерфейс	1	1	1
Наличие готовых форм	1	1	1
Возможность сохранять данные в БД	0,5	1	0,5
Простота использования	0	0	0,5
Средний бал	0,6	0,7	0,7

1.4 Постановка задачи на разработку системы

Проектирование системы способствует эффективной работе, использующейся в сфере деятельности оказания услуг. Именно качественное проектирование обеспечит создание системы, способной функционировать

при постоянном совершенствовании технических, информационных и программных составляющих.

Для эффективного решения проблемы по устранению неисправностей по поступившим заявкам необходимо спланировать работу сотрудников мастерской и взаимодействия с клиентами.

Функциональность АИС должна обеспечивать выполнение наиболее востребованных операций, таких как [10]:

– возможность вводить данные о заявках и оборудовании в созданные формы и сохранять их в базу данных;

– возможность отправлять оповещение клиенту;

– формирование отчетов по выбранным заявкам;

– возможность отслеживать статус выполнения заявки;

– просмотр списков: сотрудники, оборудование и заявки.

Требования к АИС:

– надежность и простота обслуживания;

– графический интерфейс;

– наличие готовых форм для заполнения;

– наличие базы данных для хранения информации;

– ежедневное резервное копирование данных на сторонний носитель.

Выводы по первой главе

В первой главе проведено исследование деятельности мастерской по ремонту электронной техники. Проведено моделирование бизнес-процессов, выявлены проблемные моменты, которые могут быть решены внедрением CRM-системы.

Глава 2 Проектирование информационной системы

2.1 Модели и методологии проектирования и разработки АИС

На протяжении длительного времени процесс разработки ПО осуществлялся в соответствии с методиками, наработанными в инженерной области, – стандартная практика поэтапного создания продукта, начиная с составления спецификаций и заканчивая поставкой заказчику готового ПО. Существуют стандарты ГОСТ (Россия) и ISO (Европа, Россия), CMM (Capability Maturity Model – распространен в США), регламентирующие данный процесс [11].

Рассмотрим самые распространенные и используемые типы моделей жизненного цикла ПО.

Одной из первых стала применяться каскадная модель (Waterfall), в которой каждая работа выполняется один раз, т.е. делается предположение, что каждая работа будет выполнена настолько тщательно, что после ее завершения и перехода к следующему этапу возвращения к предыдущему не потребуется. Модель прославилась тем, что на этапах выполнения проекта фиксируются каждый законченный этапам, затем приступают к выполнению следующего [12].

Работа с применением гибкой методологии состоит из серии коротких циклов (итераций), длительностью 2-3 недели. Каждая итерация включает в себя этапы планирования, анализа требований, проектирование, разработку, тестирование и документирование. По завершению каждой итерации команда предъявляет заказчику «осязаемые» результаты работы, например, первичную версию продукта или часть функционала, которую можно посмотреть, оценить, протестировать, а потом доработать или скорректировать.

Для чего нужна методология гибкой разработки?

Ускорение вывода продукта на рынок. Если вы хотите что-то сделать быстрее, нужно делать это в соответствии с Agile.

Управление изменениями в приоритетах. Если говорить про коммерческую разработку, то проблема в том, что, программисты, аналитики и дизайнеры, никогда не знают, что нужно не только заказчику, который платит.

Улучшение взаимодействия между IT и бизнесом. Это головная боль, особенно для крупных компаний, ведь у бизнеса периодически меняются требования, каждый говорит на своем языке. В результате стороны друг друга не понимают.

Манифест гибкой методологии определяет четыре основные ценности и 12 принципов для методологий, базирующихся на нем.

Ценности:

– люди и взаимодействие важнее процессов и инструментов;

– работающий продукт важнее исчерпывающей документации;

– сотрудничество с заказчиком важнее согласования условий контракта;

– готовность к изменениям важнее следования первоначальному плану.

Методику Scrum чаще всего применяют команды разработчиков приложений, но принципы и опыт ее использования можно применить к командной работе любого рода. Это одна из причин такой популярности методики. Scrum часто представляют как платформу для управления проектами по методике Agile. Участники команды Scrum проводят собрания, используют специальные инструменты и принимают на себя особые роли, чтобы организовать работу и управлять ею [13].

Несмотря на то, что первоначально метод Scrum был рассчитан на разработку IT-проектов, сегодня он применяется и в других областях. При этом он ориентируется не столько на процесс управления, сколько на сам процесс разработки. Таким образом, Scrum-управление может как дополнить собой любой другой управленческий процесс, так и выступать в качестве самостоятельного.

Инкрементная модель подходит для быстрого выпуска проекта, но при этом точное ТЗ должно быть прописано уже на старте [14].

Спиральная модель (Spiral Model), принципиальной особенностью является создание ПО не сразу, как в случае каскадного подхода, а по частям с использованием метода прототипирования. Под прототипом понимается действующий программный компонент, реализующий отдельные функции и внешние интерфейсы разрабатываемого программного обеспечения. Создание прототипов осуществляется в несколько итераций, каждая из которых соответствует созданию фрагмента или версии ПО, на ней уточняются цели и характеристики проекта, оценивается качество полученных результатов, и планируются работы следующей итерации.

Преимуществами спиральной модели являются:

– ускорение разработки за счет прототипирования;

– постоянное участие заказчика в процессе разработки;

– разбиение большого объема работы на небольшие части;

– снижение риска.

К недостаткам относятся:

– разработка длится долго и стоит дорого;

– риск застрять на начальном этапе – бесконечно совершенствовать первую версию продукта и не продвинуться к следующим.

На основе итеративной модели была создана Agile – не модель и не методология, а скорее подход к разработке. Agile («эджайл») переводится с английского как «гибкий». Включает в себя практики, подходы и методологии, которые помогают создавать продукт более эффективно. Рассмотрим наиболее популярную на сегодняшний день методологию разработки ПО – Kanban [15].

Канбан – система, построенная на визуализации процесса выполнения задач команды. Основная идея в этой системе уменьшать количество задач, выполняющихся в данный момент, на первом месте задачи. Несмотря на

недостатки, такие как, нестабильный список задач, сложность применения на долгосрочных проектах, отсутствие жестких дедлайнов, хорош для проектов, которые в стадии поддержки, где основной функционал уже разработан и остались минимальные доработки и багофиксинг. Преимуществами данной методологии являются простота в использовании и гибкость в разработке [16].

Для разработки нашей АИС выбрали именно методологию Канбан, так как в этой методологии нет ограничений по времени либо они очень широкие, еще есть немаловажный плюс это доработка ситсемы на любом из этапов, при необходимости внести какие-то изменения в интерфейс пользователя или базу данных, это можно выполнить в любое время на этапе разработки. Так как у нас есть цель, которую мы хотим достигнуть, разобьем ее на задачи и будем последовательно их выполнять. Перед нами было поставлено несколько задач, выбрать модель и методологию проектирования и разработки АИС, выбрать инструменты для разработки АИС, спроектировать программную структуру АИС, провести инфологическое проектирование. Сначала изучим самые популярные и широко применяемые методологии проектирования и выберем для нашей системы наиболее оптимальный вариант, затем изучим и выберем инструменты для разработки информационной системы. На конечном этапе можно будет выполнить инфологическое проектирование базы данных [17].

2.2 Логическая модель информационной системы

Основываясь на требования к АИС и функциональность процессов мастерской, появилась возможность построения нижеперечисленных UML – диаграмм для описания структуры АИС в формальном виде.

Диаграмма вариантов использования – диаграмма, демонстрирующая отношение между пользователем и прецедентами. Таким образом, АИС представляется на рисунок 8.

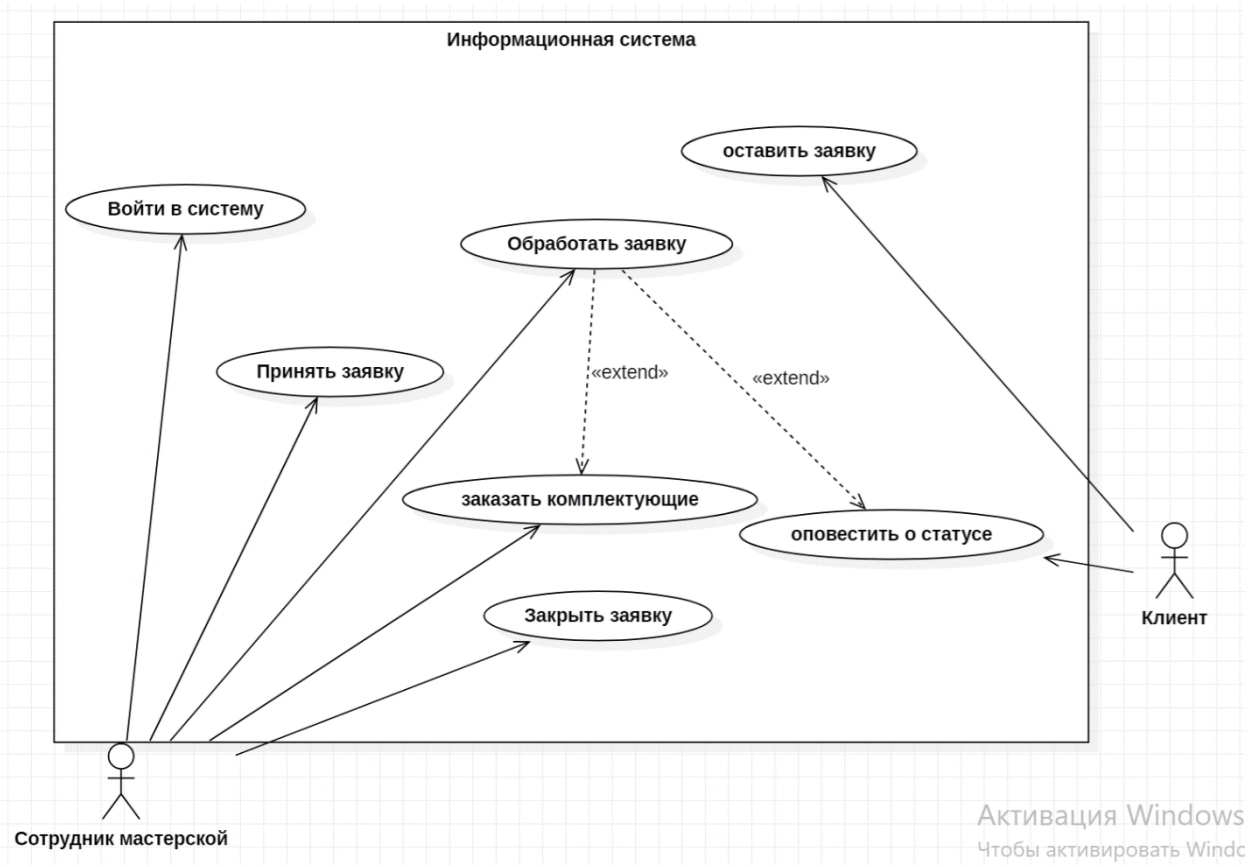

Рисунок 8 – Диаграмма вариантов использования системы

Диаграмма компонентов – статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи между компонентами. Данные диаграммы полностью описывают все варианты работы в системе, показывают работу между классами АИС и их отношения друг другу, рисунок 9.

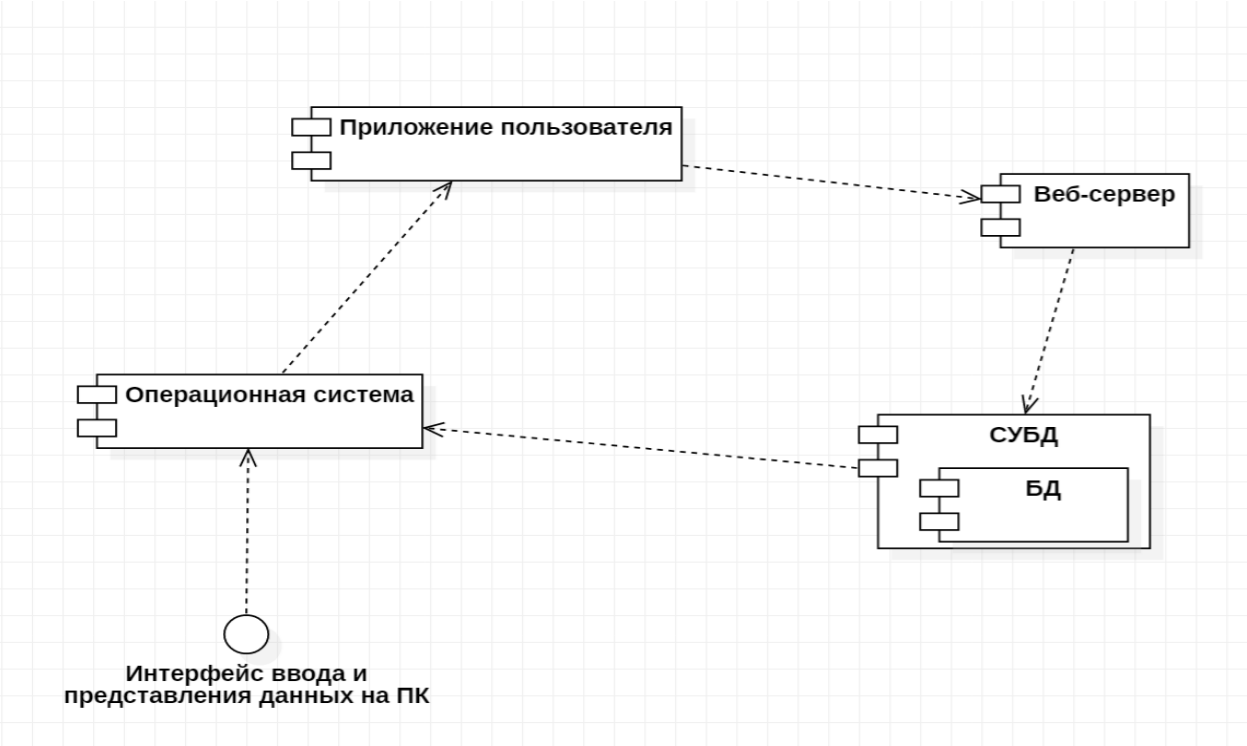

Рисунок 9 – Диаграмма компонентов

На основе информационно-логической модели АИС можно разрабатывать базу данных.

2.3 Проектирование базы данных информационной системы

Проведем инфологическое проектирование структуры базы данных.

Сущности:

Должность: Код должности, Название.

Список мастеров: Код мастера, Код должности, Фамилия, Имя, Отчество, Адрес, Телефон, Дата приема на работу.

Заказ: Код заказа, Код оборудования, Код клиента, Код мастера, Код статуса заказа, Дата заказа, Код заказа комплектующих.

Оборудование: Код оборудования, Наименование, Серийный номер, Модель.

Статус заказа: Код статуса, Наименование.

Клиент: Код клиента, Фамилия, Имя, Отчество, Телефон.

Заказ комплектующих: Код заказа комплектующих, Заказ, Выполнение.

Склад комплектующих: Код детали, Наименование, Количество, Стоимость.

Связи:

Оборудование входит в Заказ (1/O: M/O). Данный вид связи используется, так как Оборудование обязательно должно входить в заявку, а заявка обязательно должна содержать данные об оборудовании.

Заказ составляется Клиентом (M/O: 1/H). Данный вид связи используется, так как Заказ обязательно составляется Клиентом, а Клиент не обязательно может делать заявку.

Список мастеров выполняют Заказы (1/H: M/O). Данный вид связи используется, так как Список мастеров необязательно может выполнять Заказ, а Заказ обязательно должен кем-то выполняться.

Статус заказов принадлежит Заказы (1/O: M/O). Данный вид связи используется, так как Статус заказа обязательно должен принадлежать Заказу, а Заказ обязательно должен иметь статус.

Диаграмма ER-уровня модели со слабыми сущностями представлена на рисунке 10.

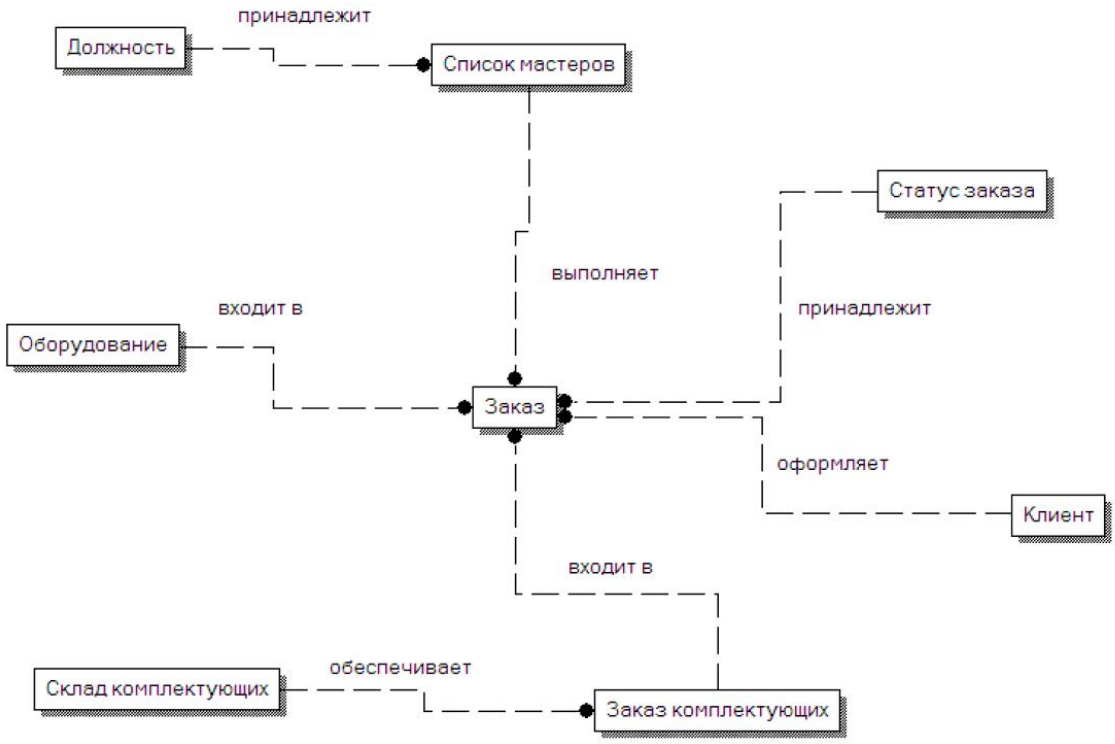

Рисунок 10 – ER-модель данных

База данных проектируется путем нормализации собранных при анализе информационных потоков данных.

Таблица 3 — Список информационных элементов

Название	Пояснение	Значение
Клиенты	Мужчина или женщина	
Код клиента	Идентификационный номер	Число. 1,2,3…
Фамилия	Фамилия клиента	Строка. Иванов
Имя	Имя клиента	Строка. Сергей
Отчество	Отчество клиента	Строка. Сергеевич
Телефон	Телефон клиента	Строка. 8-921-210-20-50
Мастера	Мужчина	
Код мастера	Идентификационный номер	Счетчик. 1,2,3…
Должность	Должность мастера	Строка. Инженер-схемотехник

Продолжение таблицы 3

Название	Пояснение	Значение
Фамилия	Фамилия мастера	Строка. Петров
Имя	Имя мастера	Строка. Игнат
Отчество	Отчество мастера	Строка. Петрович
Адрес	Адрес мастера	Строка. Г. Прокопьевск, ул. Строителей, 5 кв.2
Дата трудоустройства	Дата трудоустройства мастера	Строка. 25.10.2010
Телефон	Телефон мастера	Строка. 8-212-251-36-36
Должность	Должность мастера	
Код должности	Идентификационный номер	Счетчик. 1,2,3…
Название должности	Название должности	Строка. Инженер-схемотехник
Выполнение заказа	Информация о выполнении заказа	
Код выполнения	Идентификационный номер	Счетчик. 1,2,3…
Вид работы	Вид выполняемой работы	Строка. Замена контроллера
Стоимость работы	Стоимость	Число. 2500,3500…
Стоимость комплектующих	Стоимость	Число. 5300, 10000…
Полная стоимость заказа	Стоимость полного заказа	Число. 12500
Дата выполнения	Дата выполнения заказа	Строка. 25.06.2020
Заказ комплектующих	Заказ недостающих деталей	
Код заказа комплектующих	Идентификационный номер	Счетчик. 1,2,3…
Код детали	Идентификационный номер детали	Число. 1,2,3…
Код выполнения	Идентификационный номер выполнения	Число. 1,2,3…
Оборудование	Детали, подлежащие ремонту	
Код оборудования	Идентификационный номер	Счетчик. 1,2,3…
Наименование	Наименование оборудования	Строка. Материнская плата Esonic
Серийный номер	Номер оборудования	Строка. 9632140120
Модель	Модель оборудования	Строка. G31CEL2
Склад комплектующих	Склад, где хранятся детали	
Код детали	Идентификационный номер	Счетчик. 1,2,3…
Наименование	Наименование детали	Строка. Оперативная память Kingston
Количество	Количество товара	Число.12
Стоимость за штуку	Стоимость	Число. 12500
Статус заказа	Статус заказа	
Код статуса	Идентификационный номер	Счетчик. 1,2,3…
Наименование	Наименование статуса	Строка. В работе

Реляционная модель данных представлена на рисунке 11.

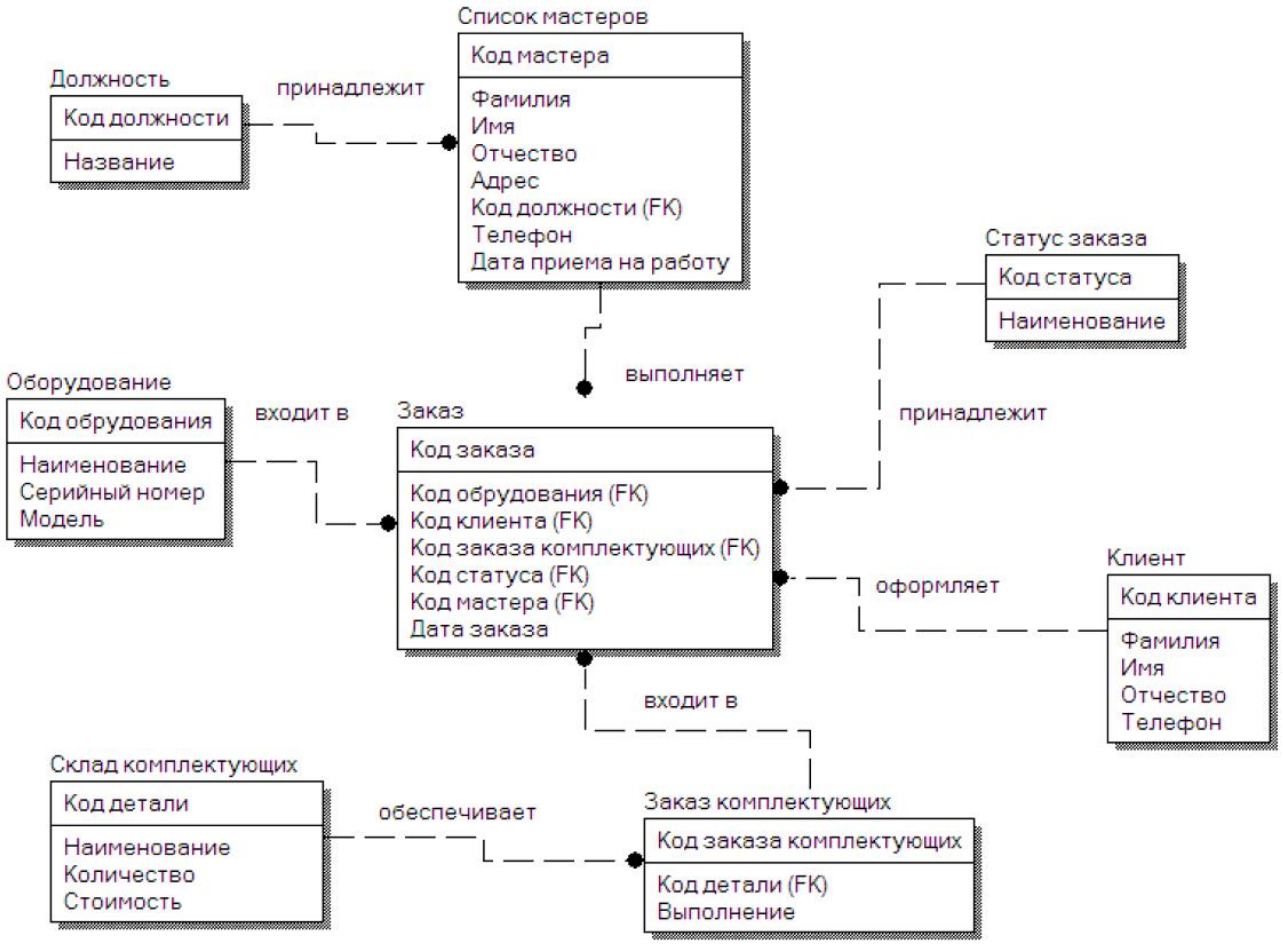

Рисунок 11 – Реляционная модель данных

2.4 Обоснование вида логической модели

Логическая модель базы данных разработана в рамках стандарта IDEF1X и выполнена средствами универсального средства Erwin Data Modeler, модель базы данных представлена на рисунке 12.

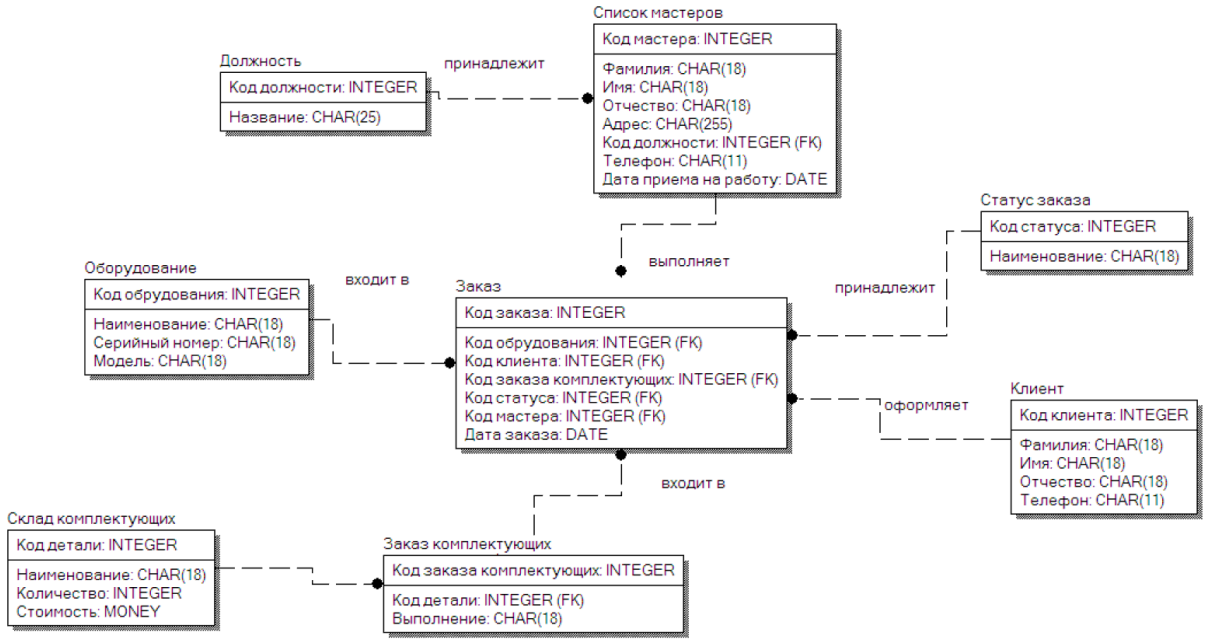

Рисунок 12 – Физическая модель базы данных

На этапе логического проектирования учтена специфика выбранной модели данных, но не учитывалась специфика конкретной системы управления базами данных.

Выполнив концептуальное проектирование, закончив разработку логической модели, следует сформулировать требования к аппаратно-программному обеспечению, определив характеристики, необходимые и достаточные для полноценного функционирования информационной системы.

2.5 Требования к аппаратно-программному обеспечению системы

В состав системы для одного филиала входят следующие технические средства:

– сервер БД;

– web-сервер;

– хранилище данных;

– АРМ мастера мастерской по ремонту электронной техники, состоящий из рабочей станции по одной на каждое рабочее место и одного на весь отдел устройства для оцифровки документов формата А4.

Сервер базы данных оснащен хранилищем данных. Хранилище поддерживает функцию резервного копирования.

Web-сервер (сервер приложений)

В состав технических средств входит сервер, имеющий следующие технические характеристики, таблица 4.

Таблица 4 – Технические средства Web-сервера

Параметр	**Значение**
Архитектура	x64
Процессор, не ниже	Intel Xeon, двухъядерный
Быстродействие процессора (тактовая частота), не менее	4,0 ГГц
Память ОЗУ, не менее	16 ГБ
Свободное место на жестком диске, не менее	140 ГБ
Пропускная способность сетевого адаптера с оптическим выходом	1 Гбит/с, не менее

На Web-сервере установлено следующее программное обеспечение:

– операционная система - Windows Server;

– web-сервер «Microsoft IIS»;

– платформа .NET Framework 4.x;

– фреймворк ASP.NET MVC 3;

– сетевые протоколы TCP/IP;

– антивирусное программное обеспечение.

В таблице 5 показаны технические средства сервера базы данных

Таблица 5 - Технические средства сервера базы данных

Параметр	**Значение**
Архитектура	x64
Процессор, не ниже	Intel Xeon с поддержкой Intel EM64T, четырехъядерный
Быстродействие процессора (тактовая частота), не менее	4,0 ГГц
Память ОЗУ, не менее	16 ГБ
Свободное место на жестком диске, не менее	1 ТБ
Пропускная способность сетевого адаптера с оптическим выходом	1 Гбит, не менее
Хранилище	не менее 3 ТБ свободного места на диске

На сервере базы данных установлено следующее программное обеспечение.

– операционная система - Windows Server;

– файловая система - NTFS;

– платформа .NET Framework 4.x;

– сетевые протоколы TCP/IP.

АРМ сотрудника мастерской должен состоять из рабочей станции по одной на каждое рабочее место и, как минимум, одного устройства на весь отдел для оцифровки документов формата А4.

Рабочие станции представляют собой IBM-совместимые персональные компьютеры (ПЭВМ), включающие в себя:

– процессор Intel Pentium 4 ГГц;

– память ОЗУ не менее 8 ГБ;

– сетевой адаптер;

– операционная система: не ниже Windows 10;

– принтер, формата А4;

– устройство для оцифровки документов;

– антивирусное программное обеспечение.

Устройство для оцифровки документов должно представлять собой планшетный сканер с возможностью сканирования объемных объектов.

Требования к сканеру представлены в таблице 6.

Таблица 6 – Технические требования к сканеру

Параметр	**Значение**
Тип	Планшетный
Тип датчика	CCD
Интерфейс	USB версии не ниже 2.0
Максимальный формат бумаги	А4 и больше
Максимальный размер документа	216x311 мм и более
Разрешение, не хуже	4800x4800 dpi
Скорость сканирования (цветное) документа А4, не менее	30 с
Формат файла сканирования	JPEG и другие
Сканирование объемных объектов	Есть

На подготовительном этапе должны быть разработаны и согласованы с заказчиком Устав проекта и план-график проведения работ. Устав проекта

должен описывать технологию проведения работ и всех участников работ от заказчика и от исполнителя.

Выводы по главе 2

После проведения исследований предметной области, выделили основные бизнес-процессы в деятельности мастерской по ремонту электронной техники. На основе бизнес-процессов было проведено концептуальное проектирование и выделены узкие места, которые тормозят весь процесс от принятия заявки клиента и до ее завершения. Также предоставилась возможность выделить основные объекты информации, на основе чего были спроектированы модели данных на логическом и физическом уровнях. Выполнив анализ модели, разработали требования к аппаратно-программному обеспечению АИС.

Глава 3 Проектирование информационной системы

3.1 Архитектура информационной системы

Архитектура системы проектировалась с учетом документа «Методика разработки специального программного обеспечения ООО «Электроника Сервис»» и является трехуровневой.

Первый уровень архитектуры системы, представляющий собой терминал, собственно приложение для конечного пользователя. На первый уровень может быть вынесена простейшая бизнес-логика: интерфейс авторизации, проверка вводимых значений.

Второй уровень является сервером приложений. На этом же уровне должен быть реализован компонент, связывающий клиентские компоненты с прикладной логикой базы данных.

На третий уровень выносится сервер базы данных, обеспечивающий хранение данных. Сервер базы данных должен представлять собой реляционную и/или объектно-ориентированную СУБД вместе с хранимыми процедурами, триггерами. [10].

Основными техническими элементами системы являются:

– web сервер;

– сервер базы данных;

– автоматизированные рабочие места. Доступ пользователей к данным CRM-системы осуществляется посредством интернет-браузера.

В качестве сервера базы данных используется «Microsoft SQL Server», в качестве сервера приложений (web-сервера) – «Microsoft IIS», операционная система - «Microsoft Windows Server». Разработка web -приложения выполнена по технологии «Microsoft ASP.NET» с использованием языка программирования «C#».

В качестве инструментального средства разработки использовалась программа «Microsoft Visual Studio 2019».

Архитектура приложения реализована для web-приложения на платформе «Microsoft .NET Framework» с помощью архитектурного шаблона Model–View–Controller (MVC), подразумевающего под собой разделение приложения на следующие три компонента.

3.2 Выбор инструментов разработки

Для автоматизированной информационной системы мастерской по ремонту электронной техники потребуется база данных для хранения информации. Рассмотрим несколько БД и выберем наиболее подходящую.

Для сравнения возьмем наиболее распространенные базы данных такие как: SQL Server, базу данных Access и PostgreSQL.

СУБД Access проста в изучение и эксплуатации и поэтому доступна для пользователей с низкой квалификацией, снабжена обширными средствами по созданию отчетов различной степени сложности, создаваемых на основе таблиц различных форматов. Как правило Access используется для создания личных баз данных, не имеющих коммерческого распространения [13].

Достоинства Access. Простота, гибкость, русификация, наличие разнообразных мастеров, конструкторов, надежная работа.

Эта база данных обладает несложными способами защиты с использованием пароля БД (возможно применения дополнительных мер по защите от несанкционированного доступа с использованием процедур VBA), в вопросах поддержки целостности данных отвечает только моделям БД небольшой и средней сложности.

PostgreSQL предоставляет множество различных возможностей, достаточно надежна и имеет хорошие характеристики по производительности. Она работает практически на всех UNIX платформах, включая UNIX подобные системы, такие как FreeBSD и Linux. Ее можно применять на Windows NT Server и Windows 2000 Server, а для разработки

годятся даже такие системы Microsoft для рабочих станций, как ME. Кроме того, PostgreSQL свободно распространяется и имеет открытый исходный код [14].

SQL Server является надежной базой данных для любых целей, может продолжать расширяться по мере наполнения информацией, без заметного уменьшения быстродействия операций с записями в многопользовательском режиме.

SQL Server обрабатывает запросы от пользователей и только отправляет пользователю результаты запроса. Таким образом, минимальная информация передается по сети.

SQL Server является приложением базы данных при работе на .Net, новейшие разработки Microsoft [15].

Представим базы данных в сравнительной таблице 7.

Таблица 7 – Результаты сравнения баз данных

Основные отличия	Access	MS SQL	PostgreSQL
1	2	3	4
Доступность	+	+	+
Хранение большого объема данных	-	+	+
Скорость обработки операций с большими данными	-	+	-
Защита от несанкционированного доступа	-	+	+
Простота использования	+	-	-

Для написания приложения нам нужно сравнить и выбрать наиболее оптимальный для нашего приложения язык программирования. Рассмотрим C++, C# и Java, наиболее распространенные.

C++ – чрезвычайно мощный язык. Однако недостатки не подходят нам для разработки, такие как синтаксис, провоцирующий ошибки, препроцессор, унаследованный от C, очень примитивен, плохая поддержка модульности (по сути, в классическом Си модульность на уровне языка отсутствует, её обеспечение переложено на компоновщика) [16].

Так как в нашем приложении будет создаваться графический интерфейс и база данных, то этот язык нам подходит со своей функциональностью и гибкостью [17].

Язык Java зародился как часть проекта создания передового программного обеспечения для различных бытовых приборов.

Фактически, большинство архитектурных решений, принятых при создании Java, было продиктовано желанием предоставить синтаксис, сходный с C и C++. В Java используются практически идентичные соглашения для объявления переменных, передачи параметров, операторов и для управления потоком выполнением кода [18].

Приведем сравнительную таблицу 8 по языкам программирования

Таблица 8 – Результаты сравнения языков программирования

Основные отличия	C++	C#	Java
1	2	3	4
Понятный синтаксис	-	+	+
Интеграция с базой данных	+	+	+
Скорость обработки данных	+	+	-
Функциональность и гибкость языка	-	+	-
Интеграция с фрэймворками	-	+	+

На основании выполненного сравнения для реализации информационной системы выбрана система управления базами данных MS SQL [19].

3.4 Разработка физической модели данных системы

Для того чтобы хранить информацию необходимо заполнить базу данными, создать таблицы, которые нам необходимы для отчетов и заполнить их атрибутами [20].

После запуска SQL Server и ввода пароля можно создавать базу данных, для этого нужно выбрать «Database» нажать правой кнопкой и выбрать «Create» и затем «Database», выйдет окно, где пишем название нашей базы Workshop и нажимаем «ОК».

Так наша база создалась, теперь можно создавать сущности и наполнять их атрибутами. Создадим несколько сущностей: «Department», «Employee», «Equipment», «Request» и наполним их атрибутами. Пример созданных таблиц представлен на рисунке 13.

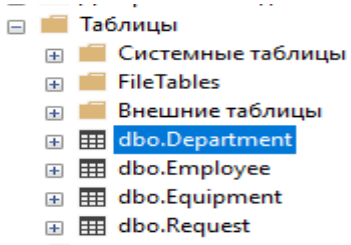

Рисунок 13 – Созданные таблицы

Наполним таблицы данными, пример заполнения данными таблицы Department показан на рисунке 14, таблицы Request на рисунке 15.

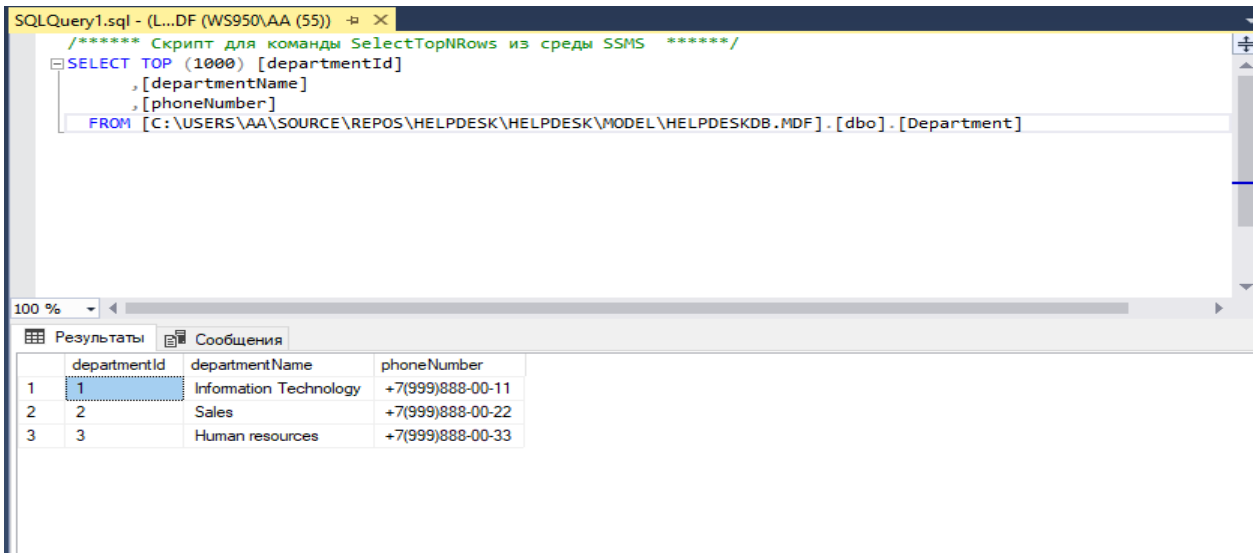

Рисунок 14 – Таблица Department

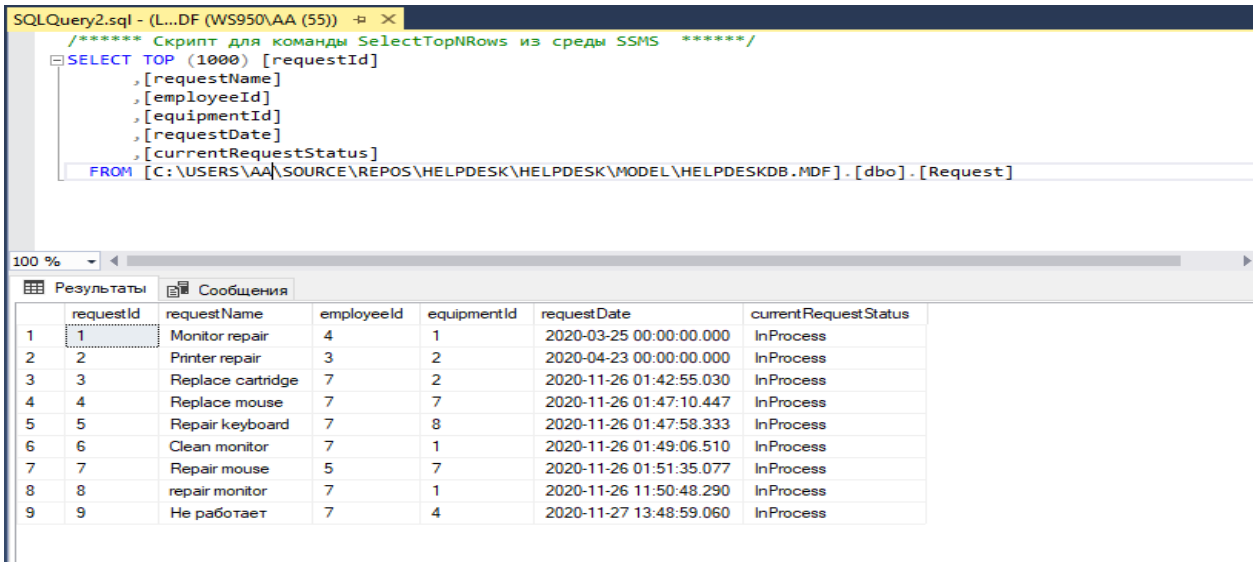

Рисунок 15 – Таблица Request

Фрагмент кода добавление новых объектов в базу данных из системы класса Employee, который будет иметь каждый свой логин и пароль, зашифрованный.

```
internal class Employee
  {
```

```
private int employeeId;
public int EmployeeId
{
   get { return employeeId; }
   set { employeeId = value; }
}
private string lastName;
public string LastName
{
   get { return lastName; }
   set { lastName = value; }
}
private string firstName;
public string FirstName
{
   get { return firstName; }
   set { firstName = value; }
}
private string middleName;
public string MiddleName
{
   get { return middleName; }
   set { middleName = value; }
}
private string position;
public string Position
{
   get { return position; }
   set { position = value; }
}
private Departament currentDepartament;
public Departament CurrentDepartament
{
   get { return currentDepartament; }
```

```
            set { currentDepartament = value; }
        }
        private string login;
        public string Login
        {
            get { return login; }
            set { login = value; }
        }
        private string passwordMd5;
        public string PasswordMd5
        {
            get { return passwordMd5; }
            set { passwordMd5 = value; }
        }
    }
}
```

3.5 Разработка программного обеспечения системы

При запуске приложения каждый пользователь вводит логин или пароль, автоматизированная информационная система выглядит как пользовательский интерфейс с кнопками, для удобства заполнения заявки на ремонт электронной техники [21]. После ввода логина и пароля необходимо пройти авторизацию, нажав кнопку «Войти».

После прохождения авторизации пользователь попадает в меню со списком оборудования, рисунок 16.

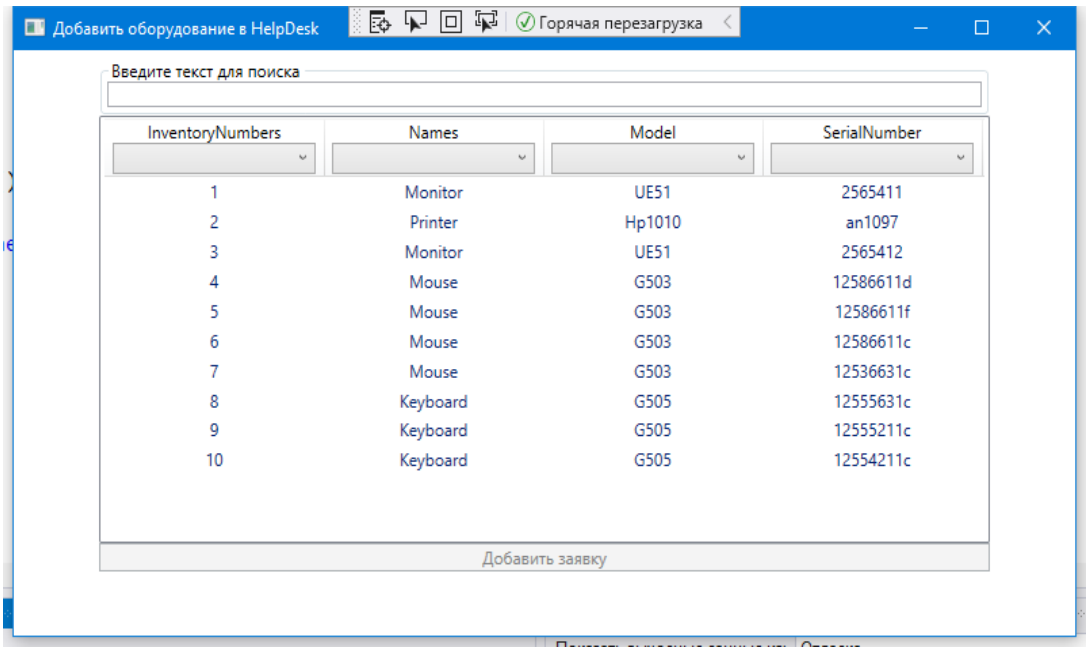

Рисунок 16 – Окно системы

В форме пользователь может выбрать оборудование из готового списка, которое нуждается в ремонте, любым способом, воспользовавшись фильтром по номеру (рис.17), по названию (рис.18), модели и серийному номеру, каждый пользователь сам определяет удобный для него способ.

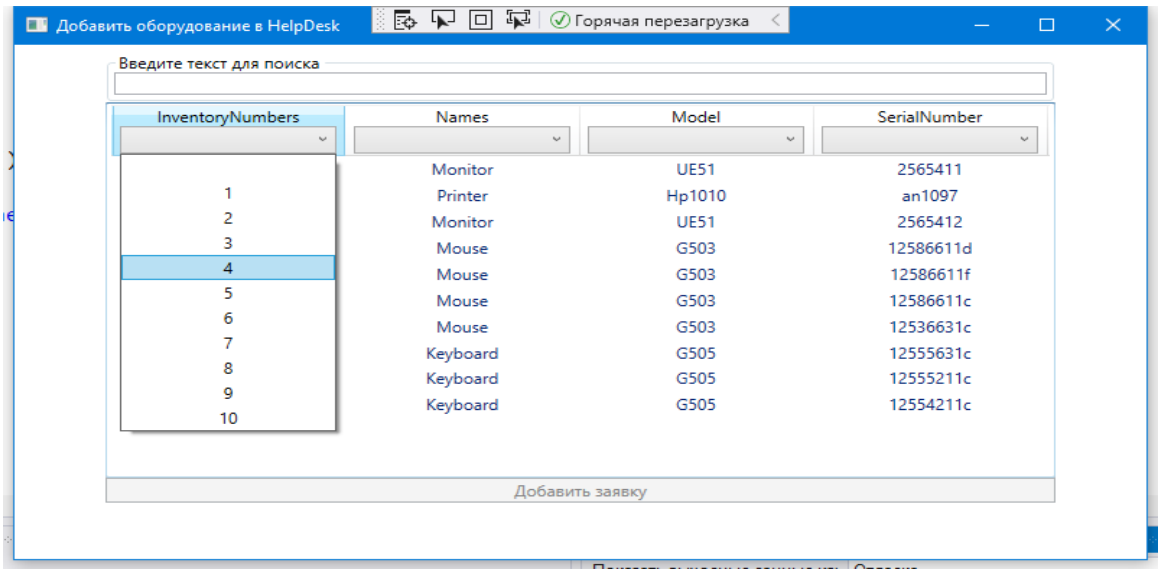

Рисунок 17 – Фильтр по номеру оборудования

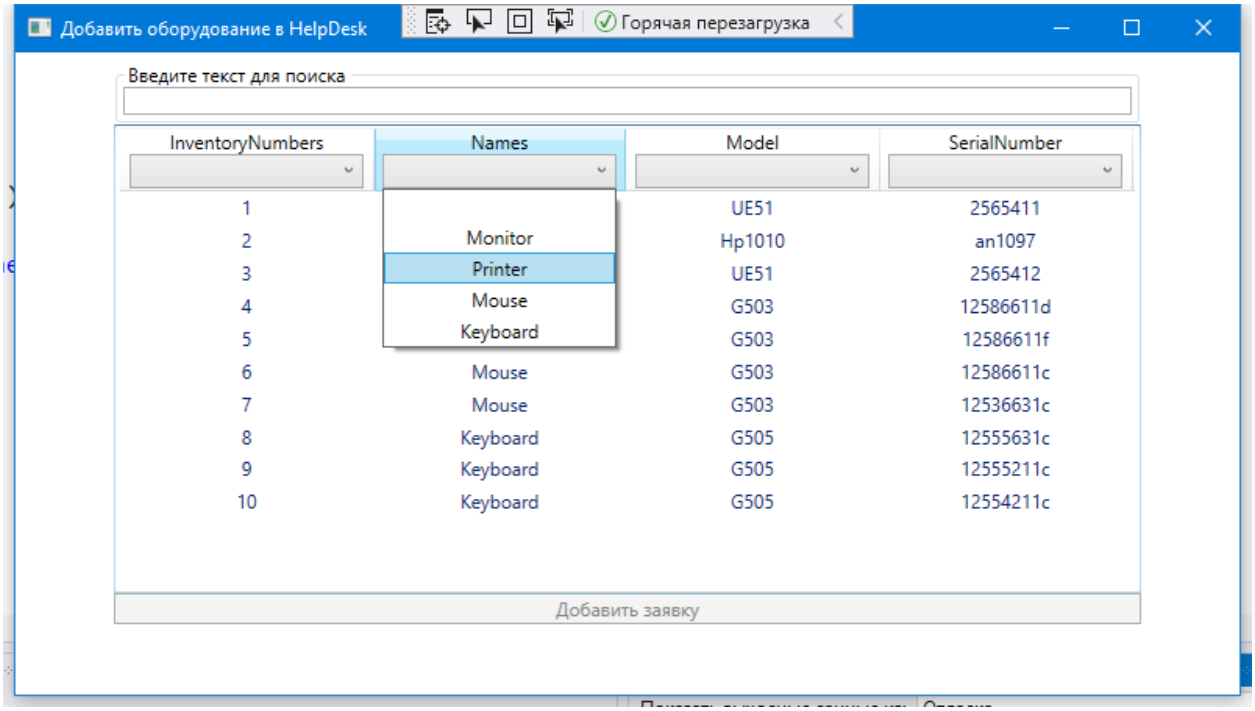

Рисунок 18 – Фильтр по названию оборудования

После выбора нужного оборудования необходимо нажать кнопку «Добавить заявку», оборудование добавится в заявку (рис.19).

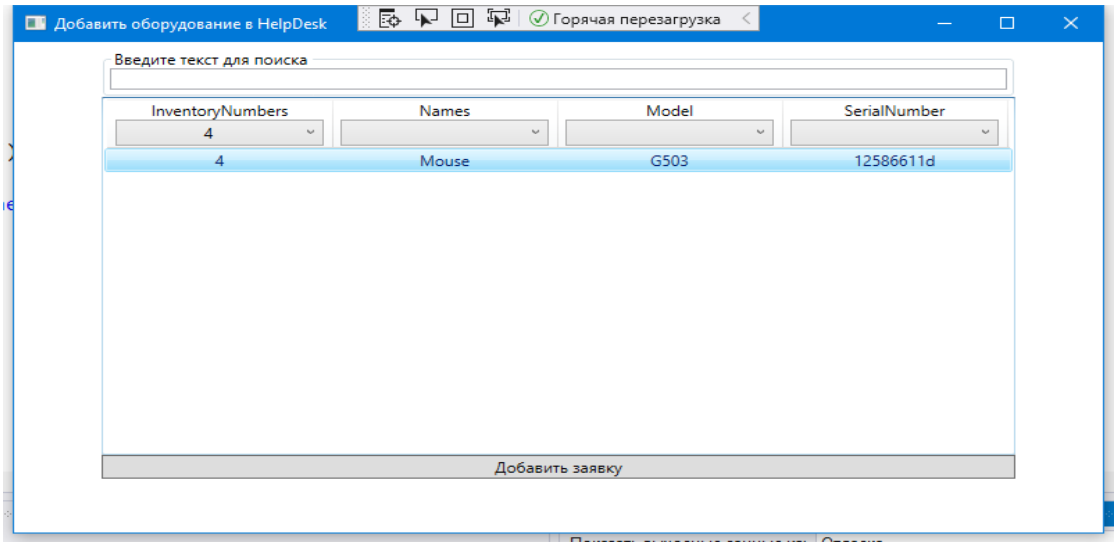

Рисунок 19 – Добавление данных в заявку

Когда все данные будут добавлены нужно нажать кнопку «Добавить заявку», после чего она сформируется в список, где можно будет добавить комментарий о поломке (рис.20).

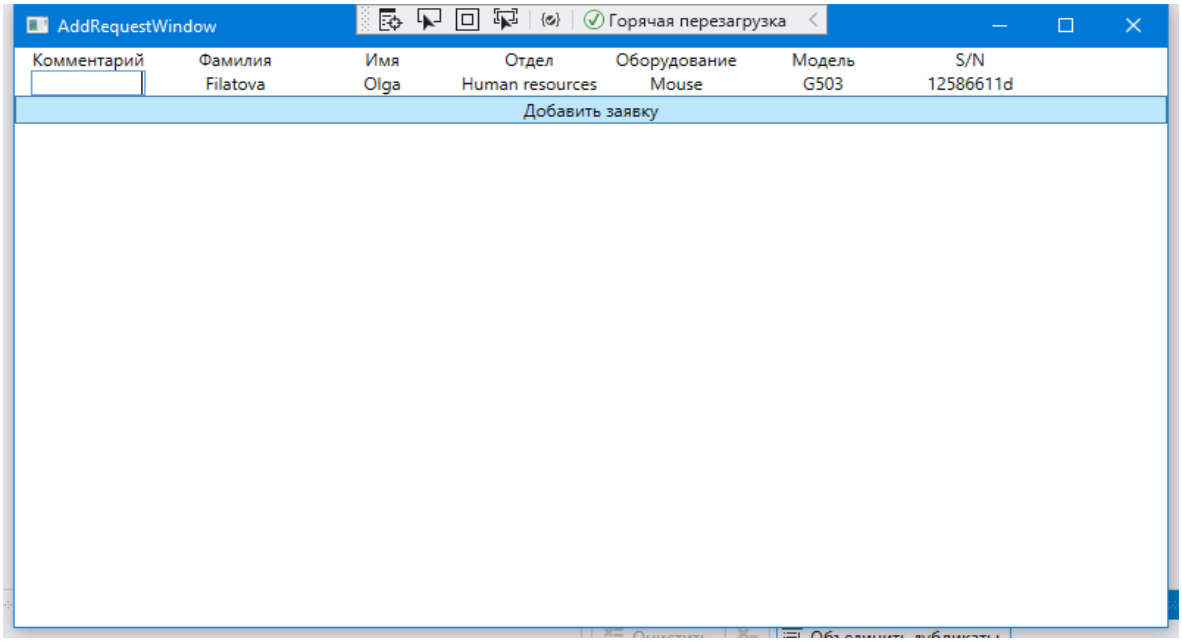

Рисунок 20 – Добавление комментария к заявке

После проверки всех данных и написания комментария к заявке при нажатии на кнопку «Добавить заявку», заявка будет добавлена в список всех заявок (рис.21), которые будет видеть сотрудник мастерской, пользователь, отправивший заявку, не будет иметь доступ к списку всех заявок [22].

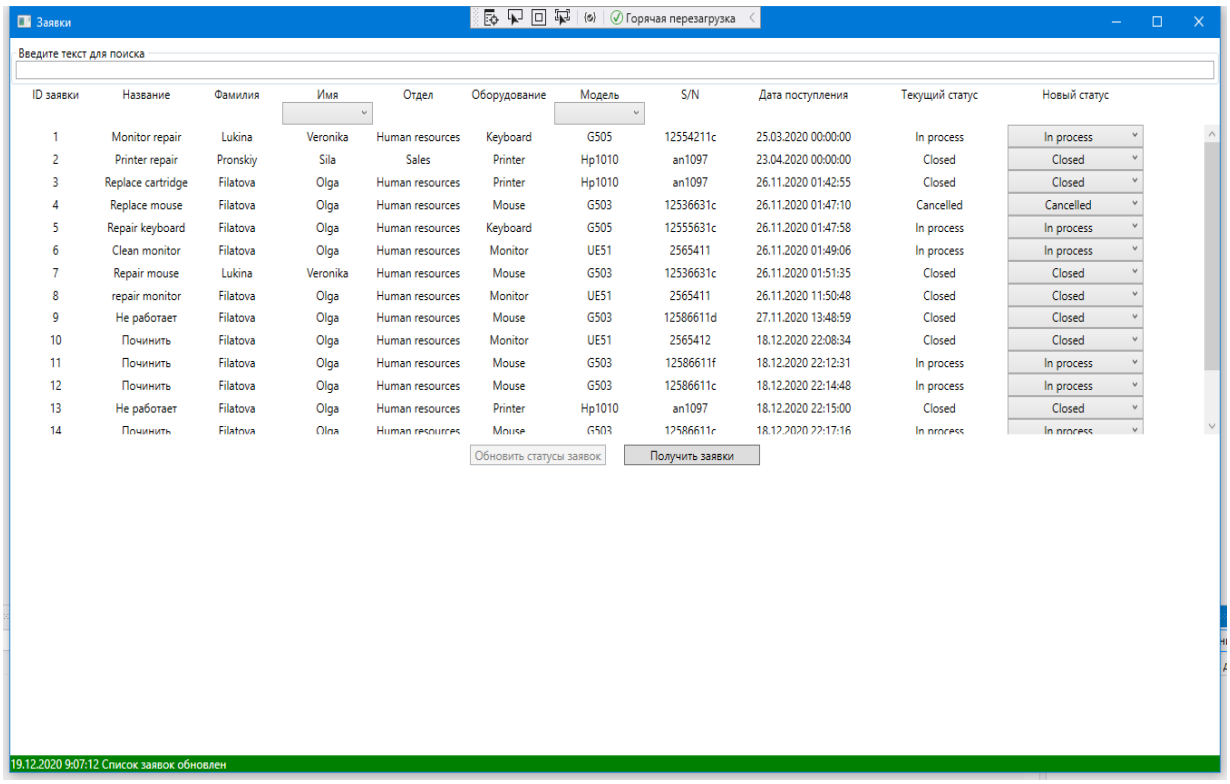

Рисунок 21 – Список заявок

Сотрудник мастерской будет иметь возможность управлять заявками путем проставления статуса заявки, клиент, подавший заявку, будет видеть у себя только оповещение о приеме заявки, которые он подал, также сможет в любое время отследить статус заявки [23].

3.6 Руководство пользователя

Для того, чтобы начать работать в приложении, нужно запустить программу. После запуска на экране появится окно регистрации, где пользователю нужно ввести свои логин и пароль. После того как будут введены данные для регистрации, система проверит и на идентификацию и даст доступ в систему, если все данные введены правильно, и такой пользователь есть в базе данных, иначе необходимо обратиться у администратора для добавления данных пользователя в базу данных.

При нажатии кнопки «Войти», пользователь попадает в окно заполнения заявки на ремонт оборудования, где он сможет выбрать из списка название, модель, серийный номер, номер оборудования.

При нажатии кнопки «Сохранить» пользователь подтверждает, что хочет добавить заявку на ремонт.

После того как пользователь подтвердил добавление заявки она сохраняется в общий список заявок, который периодически обновляется, и сотрудники мастерской могут видеть вновь поступившие заявки на ремонт.

Кнопка «Статус заявки» позволяет пользователю посмотреть на каком этапе находится его заявка.

После завершения работы в приложении, нужно просто нажать кнопку выйти, которая выведет пользователя в окно входа в систему.

Выводы по третьей главе

В третьей главе рассмотрено описан интерфейс CRM-системы мастерской по ремонту электронной техники. Описана структурная схема разработанной системы. Описаны программные модули, которые были разработаны для реализации проекта. Представлена схема взаимосвязи программных модулей. Представлено руководство пользователя системы.

Заключение

В результате выполнения выпускной квалификационной работы были выполнены все поставленные задачи:

– проведен обзор предметной области и получена информация о бизнес-процессах предприятия;

– сформированы функциональные требования к АИС;

– проведен обзор аналогичных программных продуктов, решающих подобные задачи на основе чего была подтверждена актуальность разработки АИС;

– с помощью сравнительного анализа был выбраны ЯП C#, СУБД MS SQL Server и IDE Visual Studio 2019;

– разработана логическая и физическая модели базы данных;

– реализована база данных в СУБД MS SQL Server;

– реализованы необходимые функции пользовательского интерфейса, необходимые для работы сотрудников мастерской по ремонту электронной техники;

– разработана система и успешно протестирована, что показало соответствие разработанных функций предъявляемым требованиям.

Назначение системы:

– хранение информации о клиентах и оборудовании;

– ведение учета выполненных заказов и заказов на комплектующие;

– ведение учета оборудования на складе;

– ведение списка мастеров;

– формирование отчетов за указный период о выполненных и невыполненных заказах.

Список используемой литературы и используемых источников

1. Бистерфельд О.А. Методология функционального моделирования IDEF0 : учебно-методическое пособие / О.А. Бистерфельд ; Ряз. гос. ун-т им. С.А. Есенина. — Рязань, 2008. — 48 с.
2. Воронин Б. А. Системный анализ: методические указания по выполнению курсового проекта для студентов, обучающихся с применением дистанционных образовательных технологий / Б. А. Воронин. – Томск: ФДО ТУСУР, 2021. – 82 с.
3. Гатчин Ю.А., Климова Е.В. Введение в комплексную защиту объектов информатизации: учебное пособие. – СПб: НИУ ИТМО, 2011. – 112 с.
4. ГОСТ 20886–85. Организация данных в системах обработки данных. Термины и определения [Текст]. – М.: Стандартинформ, 2005.
5. Данько Т.П. Управление маркетингом: учебник и практикум для бакалавриата и магистратуры. – М.: Издательство Юрайт, 2016. – 512 с.
6. Емельянова Н.З. Проектирование информационных систем. – М.: Форум, 2009. – 432 с.
7. Кларис-Service Desk: Web система для работы с заявками онлайн [Электронный ресурс]. – Режим доступа: http://lpsd.claris.su/?yclid (дата обращения: 13.10.2022)
8. Моделирование бизнес-процессов: учебник и практикум для академического бакалавриата / О. И. Долганова, Е. В. Виноградова, А. М. Лобанова; под ред. О. И. Долгановой. – М.: Юрайт, 2019. – 289 с.
9. Методология Kanban: доски, принципы и возможности управления. [Электронный ресурс]. – Режим доступа: https://skillbox.ru/media/(дата обращения: 13.10.2022)

10. Моделирование систем с использованием информационных технологий: учебн. пособие / В. Г. Лисиенко, Н. Г. Дружинина, О. Г. Трофимова, С. П. Трофимов. – Екатеринбург: УГТУ-УПИ, 2009. – 440 с.

11. Онлайн-курс обучения программированию: методологии разработки. [Электронный ресурс]. – Режим доступа: https://javarush.ru/groups/posts/647-metodologii-razrabotki-po (дата обращения:13.10.2022)

12. Отличия, достоинства и недостатки базы данных PostgreSQL: что такое PostgreSQL. [Электронный ресурс]. – Режим доступа: https://oracle-patches.com/common/3214-что-такое-postgresql (дата обращения: 13.10.2022)

13. Попова-Коварцева, Д.А. Основы проектирования баз данных: учеб. пособие / Д.А. Попова-Коварцева, Е.В. Сопченко. – Самара: Изд-во Самарского университета, 2019. –112 с.

14. Проектирование реляционных баз данных: Метод. указания к курсовому проектированию по курсу "Базы данных" / Московский государственный институт электроники и математики; Сост.: И.П. Карпова. – М., 2010. – 32 с

15. Сервис выполняет три основные функции. [Электронный ресурс]. – Режим доступа https://helpiks.org/7-9261.html(дата обращения: 13.10.2022)

16. Трутнев Д. Р. Архитектуры информационных систем. Основы проектирования: Учебное пособие. – СПб.: НИУ ИТМО, 2012. – 66 с.

17. Хайруллин, Р.С. Программирование на C#: учебное пособие. / Хайруллин Р.С. – Казань: Изд-во Казан. гос. архитект.-строит. ун-та, 2017. – 159 с.

18. Чудинов, И.Л. Информационные системы и технологии: учебное пособие / И.Л. Чудинов, В.В. Осипова. – Томск: Изд-во Томского политехнического университета, 2013. – 145 с.

19. Ясенев, В.Н., Дорожкин, А.В. и др. Информационная безопасность: Учебное пособие / Ясенев В.Н., Дорожкин А.В., Сочков А.Л.,

Ясенев О.В. Под общей редакцией проф. Ясенева В.Н. – Нижний Новгород: Нижегородский госуниверситет им. Н.И. Лобачевского, 2017. – 198 с.

20.Gupta A, Dengre V, Abubakar H & Manan Shah Comprehensive review of text-mining applications in finance Financial Innovation volume6, Article number: 39 (2020) Cite this article.

21. HubEx: облачная платформа управления сервисным обслуживанием. [Электронный ресурс]. – Режим доступа: https://hubex.ru/features/inside/?yclid (дата обращения: 13.10.2022)

22. Naumen: программные продукты и решения Naumen. [Электронный ресурс]. – Режим доступа: https://www.naumen.ru/products/ (дата обращения: 13.10.2022)

23. SoftClipper: что такое SQL Server. [Электронный ресурс]. – Режим доступа: https://softclipper.net/foxpro-i-sql/sravnenie-baz-dannykh-microsoft-sql-server-i-microsoft-visual-foxpro.html (дата обращения: 13.10.2022)

Приложение А

Программный код

Файл DisplayRootRegistry.cs

```
class DisplayRootRegistry
    {

        public Dictionary<Type, Type> vmToWindowMapping = new Dictionary<Type, Type>();

        public void RegisterWindowType<VM, Win>() where Win : Window, new() where VM :
class
        {
            var vmType = typeof(VM);
            if (vmType.IsInterface)
                throw new ArgumentException("Cannot register interfaces");
            if (vmToWindowMapping.ContainsKey(vmType))
                throw new InvalidOperationException(
                    $"Type {vmType.FullName} is already registered");
            vmToWindowMapping[vmType] = typeof(Win);
        }

        public void UnregisterWindowType<VM>()
        {
            var vmType = typeof(VM);
            if (vmType.IsInterface)
                throw new ArgumentException("Cannot register interfaces");
            if (!vmToWindowMapping.ContainsKey(vmType))
                throw new InvalidOperationException(
                    $"Type {vmType.FullName} is not registered");
            vmToWindowMapping.Remove(vmType);
        }

        public Window CreateWindowInstanceWithVM(object vm)
        {
            if (vm == null)
                throw new ArgumentNullException("vm");
            Type windowType = null;

            var vmType = vm.GetType();
            while (vmType != null && !vmToWindowMapping.TryGetValue(vmType, out
windowType))
                vmType = vmType.BaseType;

            if (windowType == null)
                throw new ArgumentException(
                    $"No registered window type for argument type
{vm.GetType().FullName}");

            var window = (Window)Activator.CreateInstance(windowType);
            window.DataContext = vm;
            return window;
        }


        public Dictionary<object, Window> openWindows = new Dictionary<object, Window>();
        public void ShowPresentation(object vm)
        {
            if (vm == null)
                throw new ArgumentNullException("vm");
            if (openWindows.ContainsKey(vm))
```

```
throw new InvalidOperationException("UI for this VM is already displayed");

        var window = CreateWindowInstanceWithVM(vm);
             window.Show();
             openWindows[vm] = window;
         }

         public void HidePresentation(object vm)
         {
             Window window;
             if (!openWindows.TryGetValue(vm, out window))
                 throw new InvalidOperationException("UI for this VM is not displayed");
             window.Close();
             openWindows.Remove(vm);
         }

         public async Task ShowModalPresentation(object vm)
         {
             var window = CreateWindowInstanceWithVM(vm);
             window.WindowStartupLocation = WindowStartupLocation.CenterScreen;
             await window.Dispatcher.InvokeAsync(() => window.ShowDialog());
         }

    }
    }
```

Файл RelayCommand.cs

```
class RelayCommand : ICommand
    {
         private Action<object> execute;
         private Predicate<object> canExecute;


         public RelayCommand(Action<object> execute, Predicate<object> canExecute = null)
         {
             if (execute == null)
             {
                 throw new ArgumentNullException("execute");
             }
             this.execute = execute;
             this.canExecute = canExecute;
         }

         public bool CanExecute(object parameter)
         {
             return this.canExecute == null ? true : this.canExecute.Invoke(parameter);
         }

         public event EventHandler CanExecuteChanged
         {
             add
             {
                 CommandManager.RequerySuggested += value;
             }
             remove
             {
```

```
CommandManager.RequerySuggested -= value;
            }
        }

        public void Execute(object parameter)
        {
            this.execute.Invoke(parameter);
        }

    }
}
```

Файл ViewModelBase. cs

```
            class ViewModelBase : INotifyPropertyChanged
    {
        public ViewModelBase()
        {

        }

        public event PropertyChangedEventHandler PropertyChanged;
        public virtual void OnPropertyChanged(string propertyName)
        {
            PropertyChangedEventHandler handler = this.PropertyChanged;
            if (handler != null)
            {
                handler.Invoke(this, new PropertyChangedEventArgs(propertyName));
            }
        }
    }
   }
```

Файл AutorizationWindowViewModel.cs

```
            class AutorizationWindowViewModel : ViewModelBase
    {
        List<Employee> employees;
        Dictionary<string, string> logins;

        DisplayRootRegistry displayRootRegistry;


        public AutorizationWindowViewModel(DisplayRootRegistry displayRootRegistry)
        {

            this.displayRootRegistry = displayRootRegistry;

            employees = GetEmployees();

            logins = GetLogins(employees);

        }
```

```
private List<Employee> GetEmployees()
{
    return EmployeeService.GetAllEmployees();
}

private Visibility isvisible;
public Visibility IsVisible
{
    get { return isvisible; }
    set
    {
        isvisible = value;
        OnPropertyChanged("IsVisible");
    }
}


private string currentLogin;

public string CurrentLogin
{
    get
    {
        return currentLogin;
    }
    set
    {
        currentLogin = value;
        OnPropertyChanged("CurrentLogin");
    }
}



private string currentPasswordMd5;

public string CurrentPasswordMd5
{
    get
    {
        return currentPasswordMd5;
    }
    set
    {
        currentPasswordMd5 = value;
        OnPropertyChanged("CurrentPasswordMd5");
    }
}


#region loginCommand

private RelayCommand loginCommand;
public ICommand LoginCommand
{
    get
    {
        if (loginCommand == null)
        {
```

```
loginCommand = new RelayCommand(ExecuteLoginCommand, CanExecuteLoginCommand);
                }
                return loginCommand;
            }
        }

        private void ExecuteLoginCommand(object parameter)
        {
            bool isAuthenticationPassed = CheckAuthentication(CurrentLogin,
CurrentPasswordMd5);

            if (isAuthenticationPassed == false)
            {
                MessageBox.Show("Неправильный логин или пароль");

                return;
            }

            HelpDeskRole helpDeskRole = GetHelpDeskRole(CurrentLogin, employees);

            Employee currentEmployee = GetEmployee(CurrentLogin, employees);

            if (helpDeskRole == HelpDeskRole.HelpDeskSpecialist)
            {

                HelpDeskItWindowViewModel helpDeskItWindowViewModel = new
HelpDeskItWindowViewModel(displayRootRegistry);

                displayRootRegistry.ShowPresentation(helpDeskItWindowViewModel);
            }
            else
            {
                SelectEquipmentWindowViewModel selectEquipmentWindowViewModel = new
SelectEquipmentWindowViewModel(displayRootRegistry, currentEmployee);


                displayRootRegistry.ShowPresentation(selectEquipmentWindowViewModel);
            }

            displayRootRegistry.HidePresentation(this);

        }

        private bool CanExecuteLoginCommand(object parameter)
        {
            if (!string.IsNullOrEmpty(CurrentLogin) &&
!string.IsNullOrEmpty(CurrentPasswordMd5))
            {
                return true;
            }
            else
            {
                return false;
            }
        }

        #endregion


        #region closeCommand
```

```
        private RelayCommand closeCommand;
        public ICommand CloseCommand
        {
            get
            {
                if (closeCommand == null)
                {
                    closeCommand = new RelayCommand(ExecuteCloseCommand,
CanExecuteCloseCommand);
                }
                return closeCommand;
            }
        }

        private void ExecuteCloseCommand(object parameter)
        {
            displayRootRegistry.HidePresentation(this);
        }

        private bool CanExecuteCloseCommand(object parameter)
        {
            return true;
        }

        #endregion



        #region passwordChangedCommand
        private RelayCommand passwordChangedCommand;
        public RelayCommand PasswordChangedCommand
        {
            get
            {
                return new RelayCommand(ExecChangePassword);
            }
        }

        private void ExecChangePassword(object obj)
        {
            CurrentPasswordMd5 = ((System.Windows.Controls.PasswordBox)obj).Password;
        }



        #endregion

        private HelpDeskRole GetHelpDeskRole(string login, List<Employee> employees)
        {

            Employee employee = employees.Find(x => x.Login == login);

            if (employee is null)
            {
                return HelpDeskRole.HelpDeskUser;
            }

            if (employee.CurrentDepartament.DepartmentName == "Information Technology")
            {
                return HelpDeskRole.HelpDeskSpecialist;
```

```
}
            else
            {
                return HelpDeskRole.HelpDeskUser;
            }
        }

        private Employee GetEmployee(string login, List<Employee> employees)
        {
            Employee employee = employees.Find(x => x.Login == login);

            return employee;
        }

        private bool CheckAuthentication(string login, string password)
        {
            bool result = false;

            if (logins.TryGetValue(login, out string passwordMd5))
            {
                string calculatedPasswordHash = CalculateHash(CurrentPasswordMd5);

                if (calculatedPasswordHash == passwordMd5)
                {
                    result = true;
                }
            }

            return result;
        }

        private string CalculateHash(string text)
        {
            byte[] hash = Encoding.ASCII.GetBytes(text);
            MD5 md5 = new MD5CryptoServiceProvider();

            byte[] hashEncoded = md5.ComputeHash(hash);
            string result = "";
            foreach (var oneByte in hashEncoded)
            {
                result += oneByte.ToString("x2");
            }

            return result;
        }


        private Dictionary<string, string> GetLogins(List<Employee> employees)
        {

            Dictionary<string, string> result = new Dictionary<string, string>();

            foreach (var employee in employees)
            {
                result.Add(employee.Login, employee.PasswordMd5);
            }

            return result;
        }
```

```
}

    enum HelpDeskRole
    {
        HelpDeskSpecialist,
        HelpDeskUser
```