**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 6](#_Toc73992775)

[**ГЛАВА I. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ.** 9](#_Toc73992776)

[1.1. Актуальность выбранной темы. 9](#_Toc73992777)

[1.2. Основные этапы создания хелп-деска. 11](#_Toc73992778)

[**ГЛАВА II. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ДИПЛОМНОГО ПРОЕКТА.** 13](#_Toc73992779)

[2.1. Среда разработки Microsoft Visual Studio. 13](#_Toc73992780)

[2.2. Облачные службы Google Cloud Platform. 20](#_Toc73992781)

[2.3. Служба Google Cloud SQL. 21](#_Toc73992782)

[**ГЛАВА III. ЭКСПЛУАТАЦИЯ И ЭКОНОМИЧЕСКИЙ РАСЧЕТ.** 28](#_Toc73992783)

[3.1. Руководство пользователю. 28](#_Toc73992784)

[3.2. Расчет затрат на создание системы и ее экономическая эффективность. 35](#_Toc73992785)

[**ЗАКЛЮЧЕНИЕ** 38](#_Toc73992786)

[**Список использованной литературы.** 41](#_Toc73992787)

[**Приложение А.**](#_Toc73992788)

# **ВВЕДЕНИЕ**

Самая популярная и, пожалуй, самая удобная методика работы в ИТ сфере – это методология IT Infrastructure Library (ITIL). Она была разработана британской правительственной организацией для упорядочивания отношений с подрядчиками ИТ услуг. Первоначально библиотека ITIL состояла из 7 книг, из которых основными были «Предоставление услуг» и «Поддержка» (Service Delivery и Service Support). В них были описаны процессы, которые составляют ядро ITIL, в том числе Incident management.

Процессы в Service Desk регламентируют все сложности, которые возникают в работе ИТ-отдела:

* Incident Management – процесс, отвечающий за быстрое решение инцидентов — неисправностей, повреждений, критических ошибок, требующих ответных действий. HelpDesk регистрирует статистику инцидентов и время их ликвидации.
* Problem Management – цель этого процесса в уменьшении количества инцидентов, поступающих в HelpDesk. Для этого выявляются и устраняются их причины.
* Change Management – процесс, который регламентирует только осмысленные изменения и согласование их реализации среди всех пользователей бизнес-сервисов.
* Release Management – процесс, которой ставит условие не нарушать работу компании во время выполнения каких-либо изменений. Процесс управления релизами выполняет наблюдение и установку обновленных версий программ и аппаратных средств через службу HelpDesk.
* Service Level Management – процесс, определяющий количество и состав задействованных сотрудников, а также качество услуг в службе HelpDesk. С его же помощью идет мониторинг уровня качества и проводятся операции по снижению вероятности того, что может быть оказан недоброкачественный сервис.
* Financial Management – процесс, который описывает распоряжение финансами для обеспечения деятельности других процессов.
* Availability Management – задачи, относящиеся к доступности услуг ИТ подразделения; выделяются изолированные процессы, чтобы их можно было отслеживать и делать выводы. Уровень доступности определяется постоянством, ремонтоспособностью и надежностью.
* Capacity Management – задача, отвечающая за управление ИТ активами.
* Continuity Management – контроль непрерывности ИТ сервисов. Главные направления задачи – разработка, сопровождение, реализация и проверка действий по обеспечению непрерывности деятельности бизнес-сервисов.
* Information Security Management – гарантия непрерывной безопасности сервиса и информационная надежность.

Для реализации процесса в организации и создается специальный отдел, который контактирует с сотрудниками и согласовывает ликвидацию проблем с ИТ подразделением. Этот отдел может называться Центр обслуживания пользователей (Service Desk) или Центр поддержки пользователей (Help Desk).

Задачей Help Desk является регистрация заявок пользователей, предоставление им требуемой помощи и привлечение сотрудников ИТ подразделения для скорейшего устранения проблем. Дополнительно эта служба анализирует статистику инцидентов и время их устранения. Это необходимо для оценки и повышения качества предоставления ИТ услуг.

Адекватная, продуктивная и оперативная работа любой компании, организации или сервиса зиждется на правильно подобранном программном обеспечении. Когда организация выходит на соответствующий уровень, а количество сотрудников исчисляется не по одному-два человека, а целыми сотнями, а то и тысячами, возникает необходимость в централизации их данных, сборе информации, запросов и заявок. В условиях такой массовости невозможно посадить оператора на каждую отдельную заявку – во-первых, у сотрудников компании «не хватит рук», а если и хватит – большое количество запросов будет простаивать, тем самым роняя актуальность компании. Поэтому, чтобы ИТ специалистам было удобно и просто обрабатывать запросы, а сотрудники компании не сидели без дела, был создан сервис HelpDesk (называемая в некоторых случаях service desk) – инновационная и удобная система поддержки пользователей.

Первостепенной задачей клиентского сервиса является повышение качества обслуживания. Удовлетворённость клиентов при этом главный параметр для анализа.

HelpDesk система включает в себя возможность узнать мнение пользователя максимально простым и удобным способом, а именно отправкой запроса на оценку после закрытия заявки, вне зависимости от источника её поступления.

Функционал также позволяет сформировать отчетность в разрезе по обращениям и сотрудникам на основании положительных и отрицательных отзывов, отображая их соотношение в виде диаграммы

Цель данной дипломной работы – создание системы Helpdesk с функцией отправки почты. Выбранная тема актуальна, потому что крупные организации располагают сложной и достаточно разветвленной ИТ-инфраструктурой, непрерывное функционирование всех элементов которой на необходимом уровне является обязательным условием выполнения организацией своих задач. Поддержка этой инфраструктуры в рабочем состоянии является одной из основных функций ИТ-службы предприятия. Системы HelpDesk позволят ИТ-службам обеспечить качественное выполнение этой функции.

Объектами изучения являются создание автоматизированной системы для принятия заявок, их просмотра и закрытия.

Передо мной стоят следующие задачи:

1. Выбор языка программирования.

Язык должен быть гибким, популярным и обладать возможностью использования сторонних библиотек.

1. Выбор IDE.

После выбора языка следует выбрать среду разработки, в которой этот язык раскроет весь свой потенциал.

1. Интуитивный интерфейс.

Любой пользователь должен максимально быстро разобраться в программе.

1. Отчетность

Первостепенной задачей клиентского сервиса является повышение качества обслуживания. Удовлетворённость клиентов при этом главный параметр. HelpDesk система должна включать в себя возможность узнать мнение пользователя максимально простым и удобным способом, а именно отправкой запроса на оценку после закрытия заявки. Функционал также должен позволять сформировать отчетность в разрезе по датам, отображая их в виде таблицы.

1. Оптимизация кода

Код не должен содержать в себе элементы, которые не задействованы во время работы программы.

1. Взаимодействие с пользователями

После закрытии заявки пользователь должен получить на почту уведомление или акт выполненных работ.

Для пользователей внедрение Help Desk характеризуется повышением уровня предоставляемого сервиса и уменьшением периодов остановок из-за проблем в ИТ инфраструктуре. А для самих ИТ служб внедрение подобной организации работы становится решением проблемы, когда бизнес не может понять запросов айтишников. Появляется возможность просто обосновывать вложения в ИТ и планировать бюджет согласно потребностям в развитии бизнес-процессов. Кроме того, ИТ департамент может предоставлять актуальные и достоверные данные о своей работе и повышать доступность своих услуг для пользователей.

Использование Help Desk позволяет провести аттестацию качества всего ИТ департамента. Внедрение Help Desk позволяет понять, сколько обращений поступает в службу поддержки, как часто ломается техника и насколько качественно предоставляются услуги. На основании этой информации можно делать выводы о слабых местах компании, планировать развитие ИТ инфраструктуры и обучение специалистов. Соответственно, внедрение Help Desk систем помогает заявкам пользователей быстро поступать в обработку, а руководству контролировать работу менеджеров компании, качество их обслуживания, а также эффективность работы ИТ отдела.

Для создания графических интерфейсов с помощью платформы .NET применяются разные технологии - Window Forms, WPF, приложения для магазина Windows Store (для ОС Windows 8/8.1/10). Однако наиболее простой и удобной платформой до сих пор остается Window Forms или формы. В дипломном проекте будет использован WinForms и его основные элементы. Для создания проекта также потребуется среда разработки Visual Studio. Для создания проекта в дипломной работе будет использован Visual Studio Professional 2019.

Для создания веб-формы будет использован ASP .NET.

ASP.NET (Active Server Pages для .NET) — платформа разработки веб-приложений, в состав которой входит: веб-сервисы, программная инфраструктура, модель программирования, от компании Майкрософт. ASP.NET входит в состав платформы .NET Framework и является развитием более старой технологии Microsoft ASP.

Поскольку ASP.NET основывается на Common Language Runtime (CLR), которая является основой всех приложений Microsoft .NET, разработчики могут писать код для ASP.NET, используя языки программирования, входящие в комплект .NET Framework (C#, Visual Basic.NET, J# и JScript .NET).

Программная модель ASP.NET основывается на протоколе HTTP и использует его правила взаимодействия между сервером и браузером. При формировании страницы заложена абстрактная программная модель Web Forms и на ней основана основная часть реализации программного кода.

**ГЛАВА I. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ.**

**1.1. Актуальность выбранной темы.**

Когда пользователь или клиент сталкиваются с какой-либо проблемой (неисправностью, сбоем, просто неумением), он рассчитывает получить квалифицированную помощь в работе с программным обеспечением, персональным компьютером или офисной техникой. При этом его интересует максимально быстрое разрешение проблемы. Мало что раздражает больше, чем многократные звонки сотрудникам технической поддержки организации с целью найти там помощи — тем более, когда есть вероятность остаться необслуженным, так как деятельность технической поддержки не так хорошо регулируется.

Автоматизация бизнес-процессов, в том числе оптимизация обработки всех заявок – важный процесс, позволяющий повысить скорость и эффективность работы предприятия. В сегодняшних реалиях связь потребителей услуги с её поставщиком, либо связь представителя службы технической поддержки с клиентом, почти всегда осуществляется через программную оболочку. Это позволяет сделать процесс коммуникации более удобным для обеих сторон, но что не менее важно – сделать его прозрачным, отслеживаемым и контролируемым.

Система help desk является лучшей на сегодняшний день базой данных для хранения всей информации, связанной с обращениями клиентов, выдачей заданий специалистам. Такая система обладает простым и понятным интерфейсом, помогает следить за тем, насколько последовательно исполняются работы, вести учет инцидентов, чтобы оповещать о них администраторов и следить за тем, сколько времени и денег было потрачено на тот или иной запрос, расстановка приоритетов в заявках и т.д.

Одним словом, help desk – это многофукциональное программное обеспечение, способное во многом облегчить и ускорить работу компании, а также повысить ее рентабельность и престиж.

Такого рода системы ликвидируют проблемы, связанные с медленным отзывом IT-служб на заявки пользователей. Также улучшается качество сервисного обслуживания компаний.

Соответственно, повышается уровень безопасности IT-инфраструктуры организации, снижаются затраты и появляется возможность их спрогнозировать.

Для потребителей, внедрение Helpdesk охарактеризуется улучшением качества предоставляемого обслуживания со стороны технической подержки и сокращением периодов остановок связанных с проблемами в IT-инфраструктуре.

Для IT-служб, интеграция систем, станет решением проблемы, в ситуациях, когда бизнес не понимает запросы менеджеров IT-подразделения. Помимо этого, IT-департамент сможет предоставлять корректную и актуальную информацию о проделанной работе.

Также, внедрение систем Helpdesk и Servicedesk позволит сформировать конкретные требования к сотрудникам IT-службы и вырабатывать у них определенные навыки.

Основные особенности работы сервиса HelpDesk:

* Сервис является частью (инструментом) IT-отдела, необходимого для его коммуникации с «внешним миром», к которому относятся как сотрудники других отделов, так и клиенты компании.
* В большинстве случаев HelpDesk работает с отдельными инцидентами: запрос – обработка – решение вопроса.
* Help Desk функционирует как служба технической поддержки.

Help Desk - это новое слово в организации деятельности предприятий.[[1]](#footnote-1)

Чаще всего потребность в наличии Help Desk - системы возникает в компаниях, у которых намечается активный рост. В случае, когда система техподдержки получает десятки обращений за день и не может их обработать даже при расширенном штате – это указывает на необходимость автоматизировать и оцифровать весь процесс.[[2]](#footnote-2)

Системы HelpDesk обеспечивают:

* единую точку обращения к службе поддержки.
* удобный и понятный для пользователей механизм.
* возможность направлять запросы в службу поддержки, минуя менее эффективные способы разрешения проблем, такие, как попытки решить вопрос самостоятельно или при помощи коллег, обращение к первому попавшемуся сотруднику службы ИТ, даже если тот не обязан заниматься поддержкой.
* стандартный способ регистрации и выдачи заданий специалистам.
* контроль последовательности исполнения работ, потраченного времени и ресурсов.
* хранение базы знаний по прошлым запросам, что позволяет специалистам быстро разрешать однотипные проблемы.[[3]](#footnote-3)

Достоинства и недостатки хелп-деска.

Достоинства:

* Удобная регистрация заявок.
* Архитектура, поддерживающая работу на очень больших объемах данных (сотни и тысячи одновременных сессий).
* С помощью системы есть возможность уменьшить количество обращений в службу поддержки, а также частоту поломок техники.
* Отлично получается выявить уровень предоставляемых услуг.
* Легко получается спрогнозировать и спланировать развитие IT-инфраструктуры, поднять свою техподдержку на более высокий уровень обслуживания и таким образом сделать компанию более привлекательной для клиентов, повысить ее репутацию.
* Прямая связь с клиентом позволяет своевременно реагировать не только на вопросы, но и на предложения по улучшению и развитию, которые поступают от постоянных клиентов.

Недостатки:

* Требуюет определенных навыков программиста или администратора.

Система helpdesk является лучшей на сегодняшний день базой данных для хранения всей информации, связанной с обращениями клиентов, выдачей заданий специалистам. Такая система обладает простым и понятным интерфейсом, помогает следить за тем, как исполняются работы, вести учет оценок, чтобы оповещать о некачественной поддержке начальству.

Одним словом, Help Desk – это многофукциональное программное обеспечение, способное во многом облегчить и ускорить работу компании, а также повысить ее рентабельность и престиж.

Help Desk - это новое слово в организации деятельности предприятий так как основная задача Help Desk как сервиса — это максимально автоматизировать работу службы техподдержки пользователей, настроить ее функционирование так, чтобы позволить сотрудникам IT-отделов оперативно решать проблемы и запросы пользователей. Таким образом, Helpdesk — это единое окно, куда может обратиться пользователь со своей проблемой, которую специалисты IT-службы должны решить.

**1.2. Основные этапы создания хелп-деска.**

Все основные этапы создания хелп-деска можно представить в виде такого списка:

1. Подготовительные работы.

На первой стадии требуется определить:

* интерфейс программы на компьютерах пользователей;
* веб интерфейс для просмотра базы с заявками;
* как пользователи будут уведомлены о закрытии заявки;
* какая будет использоваться база данных;
* хостинг для веб-приложения и базы данных;
* как будут происходить бэкапы для базы данных.

2. Разработка ТЗ.

Основываясь на сведениях, указанных в опросном листе, веб-разработчики составляют техническое задание, от которого зависит итоговая стоимость и срок создания системы.

Технические задания обязательно должны быть утверждены заказчиками. Техническое задание является официальным документом для проведения исследования, которое согласовывается со всеми заинтересованными лицами. Наличие ТЗ бывает достаточным для проведения работ, но иногда полезно составить план, особенно когда исследование носит широкомасштабный характер и требуется координация исполнителей. План предполагает наличие перечня работ, сроков их выполнения, ответственных за выполнение и сметы.

3. Разработка структуры.

На этой стадии разработчики продумывают общую структуру системы. В идеале следует взять как пример решения от других разработчиков/компаний, поинтересоваться у сотрудников офиса, руководства.

4. Создание дизайна.

Главное свойство создаваемой системы – интуитивность. Любой пользователь вне зависимости от образования, пола, возраста должен будет имеет возможность подать заявку.

5. Создание приложения для ПК пользователей.

Для начала специалисты должны разработать программу для подачи заявок. Как говорилось ранее – она должна быть интуитивной и приятной глазу. Программа может содержать не только функционал по отправке заявок, но и основную информацию о компьютере и пользователе, что поможет сотруднику технической поддержки удаленно помочь пользователю.

6. Создание базы данных.

В базе данных должны будут содержаться данные о открытых заявках, закрытых, пользователях и т.д.

7. Создание веб-приложения.

Как только база данных и приложение на ПК будут готовы, нужно будет создать решение, которое позволит просматривать заявки, закрывать их, автоматически отправляя уведомление на почту пользователям, а также выгружать начальникам данные по заявкам в Excel.

8. Тестирование системы.

Так как эта система включает в себя разные подсистемы, мы должны протестировать эту систему в действии, создавая заявки и закрывая их. Таким образом мы можем выявить все недочеты как с программной стороны, так и с аппаратной, и в последствии, устранить их.

9. Установка системы.

Предпоследним этапом является установка системы заказчику. Разработчики допишут/изменят некоторые пункты меню, чтобы система была уникальна для компании, что в последствии может положительно сказаться на восприятие системы обычными пользователями.

10. Сопровождение.

Для набольшей отказоустойчивости системы должна быть организована техническая поддержка разработчиками. Также может быть рассмотрен вариант обучение ИТ специалистов в компании-заказчике для устранения незначительных неполадок или изменений пунктов меню.

# **ГЛАВА II. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ДИПЛОМНОГО ПРОЕКТА.**

# **2.1. Среда разработки Microsoft Visual Studio.**

Для создания проекта будеть использован среда разработки Microsoft Visual Studio 2019. Выбор пал именно на эту среду разработки не только из-за поддержки языка C#, но и благодаря следующим положительным аспектам:

* Встроенный Web-сервер. Для обслуживания Web-приложения ASP.NET необходим Web-сервер, который будет ожидать Web-запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web-сервера позволяет запускать Web-сайт прямо из среды проектирования.
* Поддержка множества языков при разработке. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE). Более того, Visual Studio также еще позволяет создавать Web-страницы на разных языках, но помещать их все в одно и то же Web-приложение.
* Меньше кода для написания. Для создания большинства приложений требуется приличное количество стандартного стереотипного кода, и Web-страницы ASP. NET тому не исключение. Например, добавление Web-элемента управления, присоединение обработчиков событий и корректировка форматирования требует установки в разметке страницы ряда деталей. В Visual Studio такие детали устанавливаются автоматически.
* Интуитивный стиль кодирования. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам. Применяемые Visual Studio автоматически параметры форматирования можно даже настраивать, что очень удобно в случаях, когда разработчик предпочитает другой стиль размещения скобок (например, стиль K&R, при котором открывающая скобка размещается на той же строке, что и объявление, которому она предшествует).
* Более высокая скорость разработки. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно.
* Возможности отладки. Предлагаемые в Visual Studio инструменты отладки являются наилучшим средством для отслеживания загадочных ошибок и диагностирования странного поведения. Разработчик может выполнять свой код по строке за раз, устанавливать интеллектуальные точки прерывания, при желании сохраняя их для использования в будущем, и в любое время просматривать текущую информацию из памяти.

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight[[4]](#footnote-4).

Интегрированная среда Microsoft Visual Studio берет свое начало с 1995 года, когда была выпущена первая версия продукта под таким названием. До этого в начале 1990-х гг. Microsoft выпускала отдельные продукты для поддержки программирования на языках высокого уровня:

• Microsoft Visual C++ - интерактивная среда программирования на языке Visual C++, расширении языка C++, разработанном и реализованном фирмой Microsoft. Язык Visual C++ в составе всех версий Visual Studio и ныне остается наиболее популярным и широко используемым языком программирования в мире;

• Visual Basic - объектно-ориентированное расширение языка BASIC, разработанное и реализованное фирмой Microsoft, которое сразу начало активно использоваться программистами всего мира, так как сочетало в себе простоту языка BASIC с новейшими объектно-ориентированными расширениями. Многие коллеги- программисты еще в начале 1990-х гг. рассказывали об удобстве Visual Basic и предпочитали именно на нем разрабатывать программы управления GUI;

• Microsoft Visual FoxPro - интерактивная среда программирования на языке Visual FoxPro - объектно-ориентированном языке с элементами процедурного программирования, разработанном под названием FoxBase первоначально небольшой фирмой Fox Software. Привлекательной чертой этого языка для многих пользователей стали возможности обращения непосредственно из программы на этом языке к базам данных, основанных на языке SQL, в частности, программирование SQL-запросов на языке FoxPro;

• Microsoft Visual SourceSafe - разработанная фирмой Microsoft система управления версиями исходных кодов, впоследствии интегрированная со средой Visual Studio.

Все эти продукты были объединены с 1995 г. "под одной крышей" под названием Visual Studio. Новая среда получила сразу номер версии 4.0, видимо, потому, что в отдельном виде описанные выше компоненты среды уже выпускались до этого в течение нескольких лет.

Даже в столь ранней версии среда Visual Studio обрела весьма широкую популярность. Например, фирма Sun Microsystems, которая в том же 1995 г. выпустила новый язык и технологию программирования Java, - в качестве основы для реализации своих платформно-независимых Java-библиотек Abstract Windows Toolkit (AWT) для поддержки разработки GUI для платформы Windows использовала именно среду Visual Studio и реализованный в ней язык Visual C++.

В 1990-х гг. в среде университетских программистов среда Visual Studio называлась ласковым дружеским именем "вижуалка", с ударением на первое "а". Это не случайно: среда Visual Studio уже к тому времени стала надежным партнером в разработках наших программистов, что подтверждает дружественность и удобство ее интерфейса. Для сравнения, к тому времени та же фирма Sun не имела столь же популярной среды программирования. Среда NetBeans для программирования на Java появилась немного позднее - в 1997 году, как результат работы небольшой группы пражских студентов-программистов из Карлова университета. Важную роль в широком распространении среды Visual Studio внесла подписка на MSDN (Microsoft Developers' Network), благодаря которой стало возможным получать новейшие программные разработки Microsoft.

Visual Studio 97 (5.0).

Новая (пятая) версия Visual Studio, известная также как Visual Studio 97 и под кодовым названием Boston, была выпущена в 1997 году. Ее новыми возможностями и инструментами стали Visual Studio InterDev - инструмент разработки интерактивных Web-сайтов на основе технологии ASP (Active Server Pages) - и Visual Java++ - среда программирования на Java "от Microsoft".

Технология ASP - выдающаяся разработка фирмы Microsoft в области Web-программирования, появилась в 1994-1995 гг. Предназначена для разработки активных Web-страниц - шаблонов для генерации HTML-страниц для клиентских браузеров в качестве ответов на Web-запросы. Она используется до сих пор на многих сайтах. На ее основе впоследствии была разработана технология ASP.NET, которая рассматривается в "Обзор возможностей .NET"

Что касается инструмента Visual Java++, то с ним связаны события совсем другого рода - иногда подобное, к сожалению, происходят в мире ИТ, что неизбежно при острой конкуренции компаний. Фирма Microsoft познакомилась с новой Java-технологией фирмы Sun и, в качестве ответа на нее в конкуренции с фирмой Sun, разработала собственное расширение языка Java и реализовала его в новой среде под названием Visual Java++. Однако, к сожалению, эти действия не вполне соответствовали официальному порядку реализации Java, введенному фирмой Sun. До сих пор (хотя ныне владельцем Java-технологии является фирма Oracle, этот порядок фактически не изменился) название Java является официально зарегистрированным товарным знаком. Поэтому, чтобы выпустить новую реализацию Java, необходимо приобрести лицензию у фирмы-владельца Java-технологии, и пропустить для новой реализации Java специальный набор тестов на реализацию языка, библиотек и виртуальной машины, под названием Java Compatibility Kit (JCK) для проверки соответствия реализации ведомственным стандартам Java. Только если все эти тесты полностью, на 100%, пройдут, новая реализация получает право именоваться "Java". К сожалению, для реализации Visual Java++ фирмы Microsoft тесты JCK не прошли, ввиду реализованных в системе Visual Java++ расширений языка. Дело вылилось, к сожалнению для обеих компаний, в длительное судебным разбирательство между Sun и Microsoft, которое было выиграно фирмой Sun, в результате чего продукт фирмы Microsoft был переименован в VJ++ (без явного упоминания названия Java). С моей точки зрения, подобные инциденты хотя и неизбежны, но весьма нежелательны, так как с обеих сторон в них участвуют, иногда в течение нескольких лет, выдающиеся технические специалисты, тратя на все это много времени, сил и здоровья. Однако впоследствии, в 2004 году, фирмы Sun и Microsoft пришли к мировому соглашению, заключив беспрецедентное соглашение об интеллектуальной собственности сроком на 10 лет.

Таким образом, версия Visual Studio 5.0 является интегрированной средой программирования на языках Visual C++, Visual Basic, VJ++, имеющей в своем составе также инструмент Visual Studio InterDev для разработки интерактивных Web-сайтов по технологии ASP.

Visual Studio 6.0 (1998).

• Application Performance Explorer

• Automation Manager

• Microsoft Visual Modeler

• RemAuto Connection Manager

• Visual Studio Analyzer

Visual Studio 6.0 - версия, использованная фирмой Microsoft в качестве основы для создания новой версии интегрированной среды, включающей единую многоязыковую виртуальную платформу для выполнения программ - .NET.

Visual Studio.NET (2002).

Версия Visual Studio 7.0, известная также под названием Visual Studio.NET и под кодовым названием Rainier, была выпущена в 2002 году. Наиболее важное нововведение в данной версии - реализация многоязыковой платформы .NET Framework, обеспечивающей надежное и безопасное программирование с полным контролем типов и безопасности на базе управляемого кода (managed code), общей инфраструктуры языков (Common Language Infrastructure - CLI), единого промежуточного языка (Common Intermediate Language - CIL), общей системы типов (Common Type System - CTS). Платформа .NET описана в многих источниках. На платформе .NET все компиляторы транслируют исходный код с соответствующего языка (C#, C++, Visual Basic и т.д.) в бинарный формат единого промежуточного кода CIL - постфиксную запись инструкций виртуальной машины .NET - Virtual Execution System (VES). Обеспечиваются: совместимость в рамках одного .NET-приложения кодов, откомпилированных с различных языков; единый механизм контроля типов и безопасности на основе метаданных; единая среда выполнения, включающая единый многоязыковый механизм обработки исключений.

Версия Visual Studio .NET 2002 была выпущена в четырех вариантах (изданиях): Academic, Professional, Enterprise Developer, Enterprise Architect.

Поистине замечательные изменения и дополнения были реализованы в языках, поддерживаемых в среде Visual Studio.

Прежде всего, был разработан и реализован новый язык программирования - C#, на мой взгляд - наиболее мощный, современный и полный язык программирования в настоящее время.

Значительные расширения были реализованы в языках Visual C++ и Visual Basic. Фактически, несмотря на некоторую разницу в синтаксисе, эти языки стали "равновеликими" языку C#, предоставляя почти аналогичные возможности. Новая версия языка Visual C++, совместимая с .NET, получила название Managed C++, а новая версия языка Visual Basic, совместимая с .NET, - название Visual Basic.NET.

Visual Studio.NET 2003.

В следующем, 2003 году была выпущена новая версия Visual Studio.NET под кодовым названием Everett. С точки зрения развития продукта в целом, данная версия не внесла существенных изменений: основная цель выпуска новой версии состояла в том, чтобы обеспечить обновление до версии .NET Framework 1.1. Однако весьма важным нововведением Visual Studio 2003 была реализация версии .NET для мобильных устройств - .NET Compact Framework и поддержка разработки приложений для мобильных устройств средствами Visual Studio.

Как и предыдущая версия, Visual Studio 2003 выпущена в четырех изданиях: Academic, Professional, Enterprise Developer, Enterprise Architect. Причем в "максимальном" варианте - Enterprise Architect - в Visual Studio 2003 реализована поддержка средств моделирования программ в стиле другого популярного продукта - Microsoft Visio, в том числе - поддержка UML-диаграмм, визуализирующих архитектуру приложения.

Visual Studio.NET 2005.

Следующая версия интегрированной среды Visual Studio 2005 (под кодовым названием Whidbey) вышла в 2005 году.

Основные нововведения версии касались параметризованных типов данных (generics). Они были реализованы в версии языка C# 2.0 и в .NET Framework 2.0. Другая группа усовершенствований касалась ASP.NET - развития технологии ASP, предназначенной для реализаци и активных Web-страниц, в сочетании с надежностью и безопасностью кода, присущими платформе .NET. В версии Visual Studio 2005 впервые были реализованы дополнительные виды проектов для поддержки концепции ASP.NET Web-сервисов. Отметим, что средства Web-программирования в .NET заслуживают особого внимания для изучающих эту платформу и Web-программирование в целом. Только ради одного их изучения и использования каждый программист должен хорошо знать платформу .NET и практически владеть ею.

Для нашей команды разработчиков системы аспектно-ориентированного программирования Aspect.NET версия Visual Studio 2005 сыграла особую роль. Именно для этой версии среды Visual Studio был реализован plug-in под названием Aspect.NET Framework - GUI для управления аспектами в стиле, близком к визуальному стилю самой интегрированной среды. Интерактивные возможности и удобство Aspect.NET были высоко оценены пользователями, и именно в этом варианте система Aspect.NET распространилась в 26 странах мира.

Другой известный проект, реализованный как plug-in к Visual Studio.NET 2005 - это проект Knowledge.NET, расширение языка C# средствами представления знаний (фреймами, наборами правил, онтологиями), позволяющий сочетать традиционный стиль программирования на языке C # с использованием знаний. Система Knowledge.NET использована десятками наших студентов и аспирантов при разработке интеллектуальных решений, использующих знания.

Visual Studio.NET 2008.

В 2008 году была выпущена следующая версия Visual Studio - Visual Studio.NET 2008 (Orcas). Внутренний номер версии среды Visual Studio в корпорации Microsoft - Visual Studio 9. Эти номера полезно иметь в виду, так как при описании развития Visual Studio сотрудники Microsoft (например, в блогах) часто используют эти внутренние номера версий. Версия Visual Studio 2008 предназначена для ОС Windows Vista. Она соответствует версии другого популярного продукта Microsoft Office 2007 и содержит развитые средства поддержки Web-программирования. Версия .NET Framework, соответствующая версии Visual Studio 2008 - это .NET Framework 3.5.

В версии VS 2008 введен новый визуальный дизайнер приложений, использующих Windows Presentation Foundation (WPF) - API для поддержки GUI.

В отладчике VS 2008 реализована поддержка отладки многопоточных приложений.

В версию VS 2008 впервые не была включена поддержка языка J#, - расширения языка Java фирмы Microsoft, о котором говорилось выше. Вместо этого пользователям был предложен входящий в среду инструмент Java Language Converter Assiatant (JLCA) для переноса приложений из языка и среды Java в среду .NET.

Версия VS 2008 - на данный момент последняя версия среды Visual Studio, в которой имеется поддержка внутреннего продукта Microsoft - Phoenix, инструментария для разработки оптимизирующих компиляторов и языковых процессоров. Это совместный продукт Microsoft Research и Microsoft Product Team. Phoenix используется для реализации внедрения (weaving) аспектов в целевые приложения в системе Aspect.NET, так как Phoenix имеет удобное высокоуровневое внутреннее представление - Phoenix High-Level IR (HIR), в терминах которого удобно выполнять трансформации программ. Также Phoenix содержит удобный инструмент для генерации бинарной сборки .NET по этому внутреннему представлению, который и выполняет в нашей системе Aspect.NET генерацию результирующей бинарной сборки после внедрения в нее аспектов. В настоящее время Phoenix используется для проекта Microsoft Research Midori - новой операционной системы, написанной полностью в среде .NET на managed-коде.

Visual Studio.NET 2010.

В апреле 2010 г. была выпущена версия Visual Studio 2010.

Наиболее важное нововведение в Visual Studio (внутренний номер версии - 10, кодовое название - Dev10) - появление еще одного языка - функционального языка F#. Автором языка является Дон Сайм (Don Syme) из Microsoft Research, Кембридж, Великобритания. Язык F# сочетает в себе возможности чисто функционального программирования со средствами объектно-ориентированного программирования, параллельного программирования и средствами интеграции со всеми остальными возможностями .NET Framework и Visual Studio. Это весьма перспективный язык, нашедший себе много приверженцев, в особенности среди молодых программистов.

Первоначально планировалось, что в состав Visual Studio 2010 войдет продукт Phoenix и разработанный на его основе оптимизирующий компилятор с языка Visual C++, который должен был по производительности генерируемого бинарного кода превзойти имеющуюся, прежнюю, версию компилятора Visual C++. Однако, к сожалению, эти планы не были осуществлены - команда Phoenix не уложилась в сроки, поставленные руководством проекта Visual Studio.

В "максимальной" версии - Visual Studio 2010 Ultimate - имеются средства отображения структуры проектов с помощью UML-диаграмм, а также компонента Test Impact Analysis для оценки влияния модификации исходного кода проекта на пропускаемые группы тестов, что удобно, так как позволяет избежать излишних пропусков тестов. Имеется также отладчик IntelliTrace, позволяющий запоминать историю отладки и полностью ее состояние, включая историю многопоточности (а не только стек текущего потока).

Следует отметить поддержку в Visual Studio 2010 облачных вычислений. Для этого появились новые виды проектов Microsoft Azure Project. Для их использования необходимо инсталлировать для Visual Studio 2010 специальный инструментарий - Microsoft Azure SDK.

Visual Studio 2012.

В августе 2012 года вышла версия Visual Studio 2012 под кодовым названием Metro Theme. В ней был улучшен пользовательский интерфейс, разработан новый инструмент просмотра проектов (solution explorer), улучшена поддержка цветов для выделения фрагментов кода. Как отмечает Microsoft, общий размер исходного кода для Visual Studio 2012 достиг 50 миллионов строк.

Компоненты Visual Studio.

button (Кнопка) — элемент интерфейса компьютерных программ, является метафорой кнопки в технике и, соответственно, изображается схожей с ней и выполняет аналогичные функции. При нажатии на неё происходит программно связанное с этим нажатием действие либо событие.

В широком смысле, кнопкой называют любой экранный элемент, с очерченной границей, нажатие на который приводит к некоему действию.

Кнопка может быть реализована как в графических, так и в текстовых интерфейсах.

Множество возможных разновидностей сложных кнопок бесконечно. Сложные кнопки могут быть фиксируемыми (сохранять текущее состояние после завершения нажатия), реагировать на быстрые двойное, тройное и более нажатия, включать многофазные счётные триггеры и другие более сложные устройства с большим количеством состояний.

Виды кнопок:

* Простая кнопка.

Простая кнопка имеет шесть состояний — «нажато», «отжато», «в фокусе», «не в фокусе», «разблокировано» и «заблокировано».

* Счётная кнопка.

Счётная кнопка (toggle button) может иметь два, три и более состояний, работает как многофазный счетный триггер, то есть при каждом нажатии переходит в следующее состояние по модулю счёта. В общем случае счётная кнопка не удерживается в нажатом состоянии. Состояние счётной кнопки определяется косвенно, по состоянию устройства или элементов индикации. В частном случае счёта по модулю 2 счётная кнопка может удерживаться в нажатом состоянии, тогда состояние счётной кнопки определяется положением самой счётной кнопки.

В зависимости от стиля визуального исполнения может иметь как выпуклый, так и утопленный или плоский вид. Также при «нажатии» зачастую визуально имитируется утапливание её в поверхность.

* Радиокнопка.

Кнопка, входящая в группу кнопок с зависимой фиксацией для выбора одного из предложенных вариантов, выделяется в специализированный элемент управления — радиокнопку.

Группа кнопок схожей функциональности может быть объединена в единую панель инструментов.

Для управления с клавиатуры кнопка может быть снабжена управляемым фокусом ввода: при получении фокуса клавиатурный ввод (обычно пробел) инициирует нажатие на кнопку. При смене состояния управляемой кнопкой функции программы может отображать это состояние изменением значка или надписи на своей поверхности.

В устройствах с тактильной обратной связью при пересечении курсором границы кнопки реализуется имитация тактильной отдачи, подобной той, что испытывает палец, перемещаясь на поверхность реальной кнопки.

* Комбо-кнопка.

В панелях инструментов распространена практика расширения функционала кнопок за счёт использования так называемых комбо-кнопок. Такие кнопки кроме обозначенной функции могут вызывать выпадающий список. Для этого требуется либо нажать на значок выпадающего списка, расположенный на кнопке, либо держать кнопку нажатой около секунды. Также распространён вариант, когда список раскрывается, если, удерживая кнопку, перемещать курсор. Наиболее распространённые способы использования комбинированной кнопки: выполнение действий, аналогичных нажатию на кнопку несколько раз (например, в навигационных кнопках веб-браузеров имеются списки, позволяющие перейти на несколько страниц назад или вперёд), либо изменяющих значение самой кнопки (например, в Adobe Photoshop можно выбрать из группы схожих кистей одну, которая будет отображаться на панели инструментов).

Далее я приведу основные элементы управления Windows Forms, которые были использованы для создания клиента для пользователей системы.

Элемент управления Windows Forms Button позволяет пользователю щелкнуть его для выполнения действия. При щелчке кнопки мышью элемент управления выглядит так, как будто его нажимают и отпускают. Каждый раз, когда пользователь нажимает кнопку, вызывается обработчик событий Click. Код размещается в обработчике событий Click для выполнения любых действий, которые вы выбираете.

Текст, отображаемый на кнопке, содержится в свойстве Text. Если текст превышает ширину кнопки, он будет перенесен на следующую строку. Однако он будет обрезан, если элемент управления не может соответствовать общей высоте. Свойство Text может содержать ключ доступа, позволяющий пользователю "щелкнуть" элемент управления, нажав клавишу ALT с клавишей доступа. Создание ключей доступа для элементов управления Windows Forms. Внешний вид текста определяется свойством Font и свойством TextAlign.

Label.

Windows Forms элементы управления label используются для вывода текста или изображений, которые не могут быть изменены пользователем. Они используются для обнаружения объектов в форме — для предоставления описания того, что может делать определенный элемент управления, например, или для вывода сведений в ответ на событие или процесс во время выполнения в приложении. Например, метки можно использовать для добавления описательных субтитров в текстовые поля, списки, поля со списками и т. д. Можно также написать код, который изменяет текст, отображаемый меткой, в ответ на события во время выполнения. Например, если приложению требуется несколько минут для обработки изменений, можно отобразить сообщение о состоянии обработки в метке.

TextBox.

Текстовые поля Windows Forms используются для получения ввода от пользователя или для отображения текста. Элемент TextBox управления обычно используется для редактируемого текста, хотя его также можно сделать доступным только для чтения. Текстовые поля могут отображать несколько строк, переносить текст по размеру элемента управления и добавлять базовое форматирование. Элемент TextBox управления допускает единый формат для текста, отображаемого или вводимого в элемент управления.

В формах Windows TextBox играет важную роль. С помощью TextBox пользователь может вводить данные в приложение, оно может быть одной строкой или несколькими строками. TextBox является классом и определяется в пространстве имен System.Windows.Forms.

Для отображения нескольких типов форматированного текста используйте элемент управления RichTextBox.

Текст, отображаемый элементом управления, содержится в свойстве Text. По умолчанию вы можете ввести до 2048 символов в текстовое поле. Если вы установите для свойства Multiline значение true, вы можете ввести до 32 КБ текста. Свойство Text может быть установлено во время разработки с помощью окна «Свойства», во время выполнения в коде или путем ввода данных пользователем во время выполнения. Текущее содержимое текстового поля можно получить во время выполнения, прочитав свойство Text

# **2.2. Облачные службы Google Cloud Platform.**

Поскольку ситемы такого рода требует использование баз данных, то появилась потребность в исследовании облачных сервисов, которые будут предоставлять услуги серверов. Зорошим решением будет аренда облачных серверов, так как мощности персонального компьютера может не хватить для такой обширной системы, но и расположение некоторых элементов системы на облако положительно скажется как на отказоустойчивости системы, так и на реальную цену для тех, кто в будующем захочет приобрести систему. Таким образом оптимальным решением будет Google Cloud Platform потому, что многие уже существующие сервисы, помимо сервисов Google, доверяют этой платформе.

Google Cloud Platform (рус. «Облачная платформа Google», сокр. GCP) — предоставляемый компанией Google набор облачных служб, которые выполняются на той же самой инфраструктуре, которую Google использует для своих продуктов, предназначенных для конечных потребителей, таких как Google Search и YouTube. Кроме инструментов для управления, также предоставляется ряд модульных облачных служб, таких как облачные вычисления, хранение данных, анализ данных и машинное обучение. Для регистрации нужно иметь банковскую карту или банковский счет.

Google Cloud Platform предоставляет такие услуги, как инфраструктура как услуга, платформа как услуга, и бессерверные вычисления.

В апреле 2008 года Google анонсировала App Engine — платформу для разработки и хостинга веб-приложений в дата-центрах Google. Это был первый облачный сервис, представленный компанией. Для общественности сервис стал доступен в декабре 2011 года. С момента анонса App Engine, Google успела добавить многочисленные облачные службы к своей платформе.

Google Cloud Platform является частью Google Cloud, который также включает G Suite, корпоративные версии Android и Chrome OS, а также API для машинного обучения и Google Maps.

В январе 2019 года Google запустила четыре новые программы сертификации для облачных разработчиков и инженеров: Professional Cloud Developer, Professional Cloud Network Engineer (beta), Professional Cloud Security Engineer (beta), а также G Suite. Пройти обучение можно на платформе Coursera и у других партнёров компании.

Продукты

* App Engine — платформа как услуга для хостинга приложений.
* BigQuery — инфраструктура как услуга, масштабируемая аналитика для баз данных.
* BigTable — инфраструктура как услуга, масштабируемая NoSQL база данных.
* Cloud AutoML — набор продуктов для машинного обучения, которые позволяет разработчикам с ограниченным опытом работы в области машинного обучения использовать технологии обучения и создания нейронных сетей.
* Cloud Datastore — документоориентированная облачная база данных.
* Cloud Pub / Sub — услуга для публикации и подписки на потоки данных и сообщения. Приложения могут обмениваться данными через публикацию / подписку, без прямого обмена сообщениями.
* Compute Engine — инфраструктура как услуга, предоставляет виртуальные машины.
* Kubernetes Engine — система автоматического развертывания, масштабирования и управления приложений в контейнерах для Kubernetes.
* Google Genomics — анализ геномов в облаке
* Google Video Intelligence
* Cloud Vision
* Storage — инфраструктура как услуга, предоставляет онлайн REST-доступ к файлам и содержанию хранилищ данных.

# **2.3. Служба Google Cloud SQL.**

Для удешевления процесса разработки системы и ее конечной стоимости была выбрана служба Google Cloud SQL. Эта служба специально была создана чтобы упростить процесс создания облачной базы данных. Тоесть, вместо того, чтобы создавать виртуальный сервер из образа, а только потом устанавливать базу данных. Служба Google Cloud SQL позволяет пользователям сразу устанавливать образ базы, которая будет использоваться в дальнейшем, а редактирование базы данных будет производиться на локальном компьютере через специализированное программной обеспечение, например Microsofr SQL Server Management Studio (рисунок – 2.1).

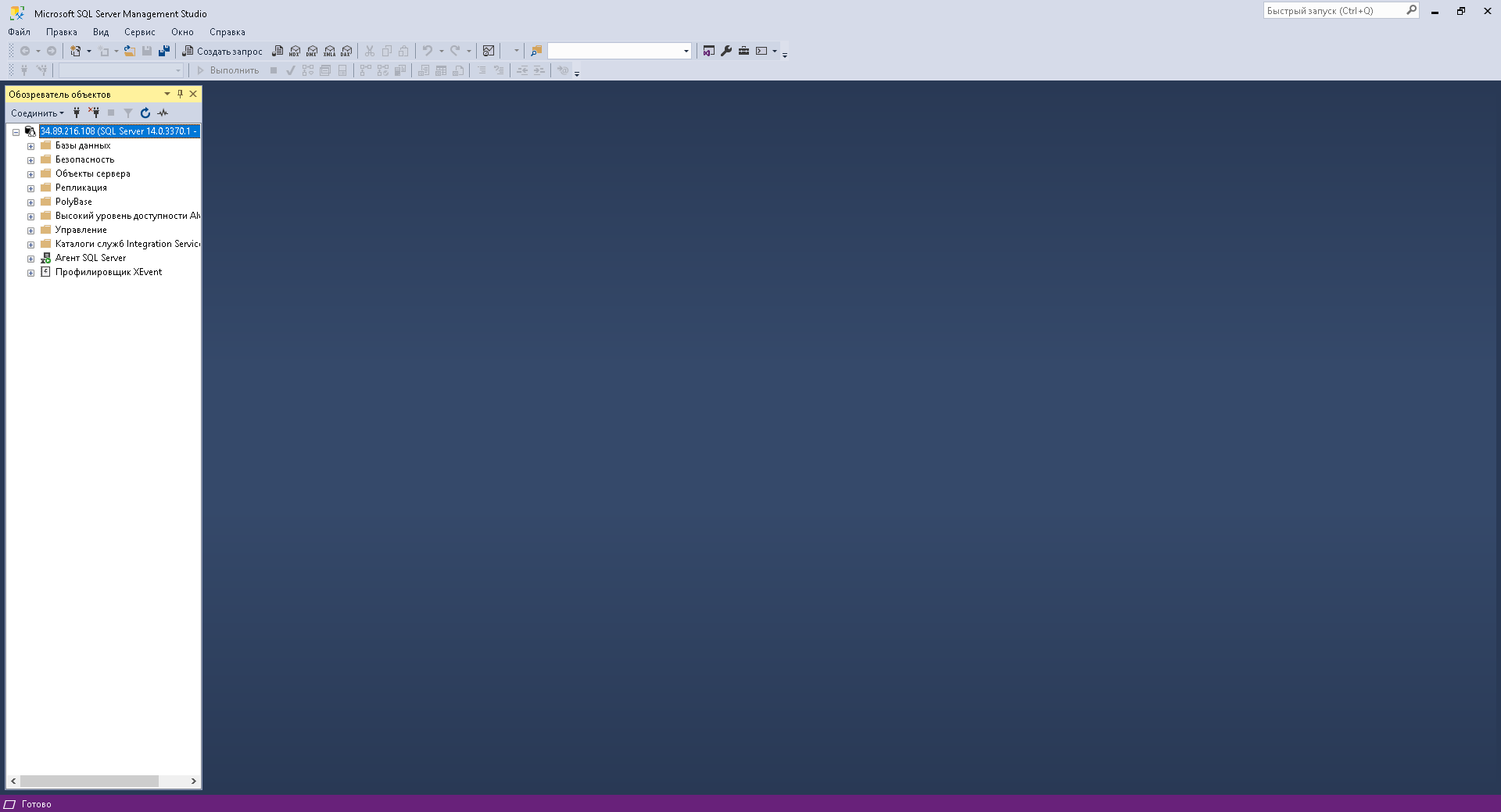


Рисунок – 2.1 - Microsoft SQL Server Management Studio.

В дипломной работе использована одна база дынных, в которой содержаться 5 таблиц с данными о сотрудниках предприятия, активными заявками, закрытыми заявками, всеми заявками и оценками от пользователей. В основном использованы типы данных text и int. Для генерации уникального номера был специфицирован идентификатор (рисунок 2.2)

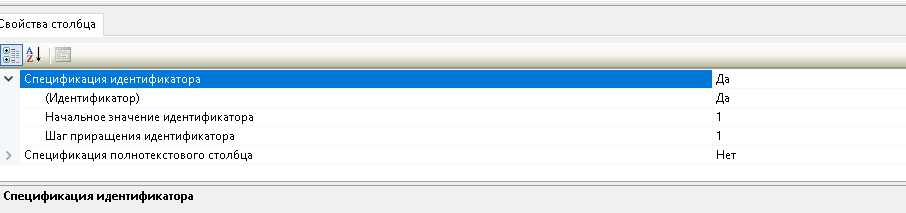


Рисунок 2.2 – Спецификация идентификатора.

Google Cloud SQL - веб-служба, которая позволяет создавать, формировать и использовать реляционные базы данных со своими приложениями на основе App Engine приложений. Это полностью управляемое обслуживание, которое поддерживает и управляет базами данных, позволяя разработчикам сосредоточиться на реализации приложений и необходимых услуг.[[5]](#footnote-5)

Основные особенности данного сервиса:

1. Легкость в использовании - богатый графический интерфейс пользователя учитывает создание, формирование, управление и контроль баз данных;

2. Полностью управляемый сервис - никаких забот о репликации, настройке и других похожих задачах базы данных. Все эти задачи выполняются сервисом;

3. Высоко доступный - чтобы соответствовать критическим потребностям доступности сегодняшних приложений и сервисов, такие функции как репликация в многократные географические области уже встроены, таким образом, сервис остаётся доступен, даже если центр данных становится недоступным.

Microsoft SQL Server.

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов.

SQL Server был создан компанией Microsoft. Первая версия вышла в 1987 году. А текущей версией является версия 2019.

SQL Server долгое время был исключительно системой управления базами данных для Windows, однако начиная с версии 16 эта система доступна и на Linux.

SQL Server характеризуется такими особенностями как:

* Производительность. SQL Server работает очень быстро.
* Надежность и безопасность. SQL Server предоставляет шифрование данных.
* Простота. С данной СУБД относительно легко работать и вести администрирование.

Центральным аспектом в MS SQL Server, как и в любой СУБД, является база данных. База данных представляет хранилище данных, организованных определенным способом. Нередко физически база данных представляет файл на жестком диске, хотя такое соответствие необязательно. Для хранения и администрирования баз данных применяются системы управления базами данных (database management system) или СУБД (DBMS). И как раз MS SQL Server является одной из такой СУБД.

Для организации баз данных MS SQL Server использует реляционную модель. Эта модель баз данных была разработана еще в 1970 году Эдгаром Коддом. А на сегодняшний день она фактически является стандартом для организации баз данных.

Реляционная модель предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта.

Для идентификации каждой строки в рамках таблицы применяется первичный ключ (primary key). В качестве первичного ключа может выступать один или несколько столбцов. Используя первичный ключ, мы можем ссылаться на определенную строку в таблице. Соответственно две строки не могут иметь один и тот же первичный ключ.

Через ключи одна таблица может быть связана с другой, то есть между двумя таблицами могут быть организованы связи. А сама таблица может быть представлена в виде отношения ("relation").

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL посредством специального API. СУБД должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

Изначально язык SQL был разработан в компании IBM для системы баз данных, которая называлась System/R. При этом сам язык назывался SEQUEL (Structured English Query Language). Хотя в итоге ни база данных, ни сам язык не были впоследствии официально опубликованы, по традиции сам термин SQL нередко произносят как "сиквел".

В 1979 году компания Relational Software Inc. разработала первую систему управления баз данных, которая называлась Oracle и которая использовала язык SQL. В связи с успехом данного продукта компания была переименована в Oracle.

Впоследствии стали появляться другие системы баз данных, которые использовали SQL. В итоге в 1989 году Американский Национальный Институт Стандартов (ANSI) кодифицировал язык и опубликовал его первый стандарт. После этого стандарт периодически обновлялся и дополнялся. Последнее его обновление состоялось в 2011 году. Но несмотря на наличие стандарта нередко производители СУБД используют свои собственные реализации языка SQL, которые немного отличаются друг от друга.

Выделяются две разновидности языка SQL: PL-SQL и T-SQL. PL-SQL используется в таких СУБД как Oracle и MySQL. T-SQL (Transact-SQL) применяется в SQL Server.

В зависимости от задачи, которую выполняет команда T-SQL, он может принадлежать к одному из следующих типов:

DDL (Data Definition Language / Язык определения данных). К этому типу относятся различные команды, которые создают базу данных, таблицы, индексы, хранимые процедуры и т.д. В общем определяют данные.

В частности, к этому типу мы можем отнести следующие команды:

CREATE: создает объекты базы данных (саму базу даных, таблицы, индексы и т.д.)

ALTER: изменяет объекты базы данных

DROP: удаляет объекты базы данных

TRUNCATE: удаляет все данные из таблиц

DML (Data Manipulation Language / Язык манипуляции данными). К этому типу относят команды на выбору данных, их обновление, добавление, удаление - в общем все те команды, с помощью которыми мы можем управлять данными.

К этому типу относятся следующие команды:

SELECT: извлекает данные из БД

UPDATE: обновляет данные

INSERT: добавляет новые данные

DELETE: удаляет данные

DCL (Data Control Language / Язык управления доступа к данным). К этому типу относят команды, которые управляют правами по доступу к данным. В частности, это следующие команды:

GRANT: предоставляет права для доступа к данным

REVOKE: отзывает права на доступ к данным

Microsoft SQL Server 2017

Программа Microsoft SQL Server 2017 может похвастаться высоким уровнем защиты данных. Она способна надежно защитить данные как во время хранения, так и при их передаче. Кроме того, именно в SQL Server было обнаружено наименьшее число уязвимостей системы безопасности, в сравнении с другими популярными СУБД, по данным Национального института стандартов и технологий США.

Кроме того, программа может похвастаться превосходным уровнем производительности, не уступающем лидерам отрасли. Это стало возможным, благодаря внедрению новых технологий работы с памятью, позволяющим реализовать новые сценарии работы и ускорить свой бизнес.

Одной из главных особенностей Microsoft SQL Server 2017 является возможность использовать новые гибридные облачные решения. Это означает, что вы сможете воспользоваться такими преимуществами, как резервное копирование данных на облако, а также аварийное восстановление локального SQL сервера. К тому же, вы сможете полностью контролировать любые данные, независимо от их места хранения.

К прочим преимуществам Microsoft SQL Server 2017 можно также отнести комплексную мобильную аналитику в реальном времени, способную поставлять до миллиона прогнозов в секунду, а также обрабатывать данные и превращать их в осмысленные отчеты, доступные на любых устройствах.

SQL Server 2017 включает множество новых функций, усовершенствований и улучшений работы для ядра СУБД.

Сборки CLR теперь можно добавлять в список доверенных в качестве обходного пути для функции clr strict security, описанной в CTP 2.0. Для поддержки списка доверенных сборок (RC1) добавлены функции sp\_add\_trusted\_assembly, sp\_drop\_trusted\_assembly и sys.trusted\_assemblies.

Возобновляемая перестройка индексов в подключенном режиме: позволяет возобновить эту операцию с момента остановки после сбоя (например, при отработке отказа в реплику или нехватке места на диске) либо приостановить и возобновить ее позже. В разделе ALTER INDEX и руководящие принципы для операций с индексами в сети.

Параметр IDENTITY\_CACHE для ALTER DATABASE SCOPED CONFIGURATION позволяет избежать пропусков в значениях столбцов удостоверений при непредвиденной перезагрузке или отработке отказа сервера на вторичный сервер. В разделе ALTER DATABASE SCOPED CONFIGURATION.

Новое поколение функций обработки запросов, использующих стратегии оптимизации для адаптации к среде выполнения рабочих нагрузок ваших приложений. В первую версию семейства функций для адаптивной обработки запросов мы включили три улучшения: адаптивные соединения в пакетном режиме, обратная связь по временно предоставляемому буферу памяти в пакетном режиме и выполнение с чередованием для многооператорных функций с табличными значениями.

Автоматическая настройка базы данных предоставляет сведения о возможных проблемах с обработкой запросов и рекомендуемые решения. Она также может автоматически исправлять выявленные проблемы.

Новые возможности для баз данных графов, предназначенные для моделирования связей "многие ко многим", включают новый синтаксис CREATE TABLE для создания граничных таблиц и таблиц узлов, а также ключевое слово MATCH для запросов.

Параметр sp\_configure, который называется clr strict security, включен по умолчанию для повышения безопасности сборок CLR.

Программа установки теперь позволяет задать для каждого файла tempdb начальный размер до 256 ГБ (262 144 МБ). Если размер файла превышает 1 ГБ, а мгновенная инициализация файлов не включена, выдается соответствующее предупреждение.

Столбец modified\_extent\_page\_count в sys.dm\_db\_file\_space\_usage отслеживает разностные изменения в каждом файле базы данных, что позволяет использовать интеллектуальные решения для полного или разностного резервного копирования, основываясь на проценте измененных страниц базы данных. Синтаксис T-SQL SELECT INTO теперь поддерживает загрузку таблицы в файловую группу, отличную от пользовательской группы по умолчанию, с помощью ключевого слова ON.

Теперь поддерживаются транзакции между всеми базами данных, входящими в группу доступности AlwaysOn, включая базы данных, являющиеся частью одного экземпляра.

Новые функции для Групп доступности включают поддержку чтения и масштабирования без кластера, параметр минимального числа реплик для фиксации, возможности миграции между Windows и Linux и тестирование на обеих системах.

Новые динамические административные представления:

sys.dm\_db\_log\_stats предоставляет сводку атрибутов и сведения о файлах журналов транзакций, которые помогают контролировать работоспособность в журналах транзакций.

sys.dm\_tran\_version\_store\_space\_usage отслеживает использование хранилища версий для каждой базы данных, что помогает оперативно планировать размеры tempdb.

sys.dm\_db\_log\_info предоставляет сведения о виртуальных файлах журнала (VLF) для отслеживания возможных проблем с журналами транзакций, оповещения об этих проблемах и их предотвращения.

sys.dm\_db\_stats\_histogram — это новое динамическое административное представление для анализа статистики.

sys.dm\_os\_host\_info предоставляет сведения об операционной системе для Windows и Linux.

Инструмент Database Tuning Advisor (DTA) получил дополнительные функции и более высокую производительность.

Усовершенствования работы в памяти включают поддержку вычисляемых столбцов в оптимизированных для памяти таблицах, а также полную поддержку функций JSON и оператор CROSS APPLY для модулей, скомпилированных в собственном коде.

Новые строковые функции: CONCAT\_WS, TRANSLATE и TRIM, а для функции STRING\_AGG теперь поддерживается WITHIN GROUP.

Появились новые параметры массового доступа (BULK INSERT и OPENROWSET(BULK...)) для файлов CSV и BLOB-файлов Azure. Улучшения работы с оптимизированными для памяти объектами включают sp\_spaceused и устранение ограничения в 8 индексов для оптимизированных для памяти таблиц, sp\_rename для этих таблиц и для скомпилированных в собственном коде модулей T-SQL, а также CASE и TOP (N) WITH TIES для скомпилированных в собственном коде модулей. Файлы файловой группы, оптимизированные для памяти, теперь можно хранить, помещать в резервную копию и восстанавливать с помощью службы хранилища Azure.

DATABASE SCOPED CREDENTIAL — новый класс защищаемых элементов, поддерживающих разрешения CONTROL, ALTER, REFERENCES, TAKE OWNERSHIP и VIEW DEFINITION. ADMINISTER DATABASE BULK OPERATIONS теперь отображается в sys.fn\_builtin\_permissions.

Службы SQL Server 2017 Integration Services (SSIS).

Новый компонент Горизонтальное увеличение масштаба в SSIS содержит новые и измененные функции.

Мастер масштабирования горизонтального увеличения масштаба теперь поддерживает высокий уровень доступности.

Улучшена отработка отказа для журналов выполнения из рабочих ролей горизонтального увеличения масштаба.

Параметр runincluster хранимой процедуры [catalog].[create\_execution] переименован в runinscaleout для согласованности и удобства чтения.

Каталог SSIS содержит новое глобальное свойство, позволяющее указать режим по умолчанию для выполнения SSIS-пакетов.

В новом компоненте Горизонтальное увеличение масштаба для SSIS вы теперь можете использовать параметр Use32BitRuntime при активации выполнения.

SQL Server 2017 Integration Services (SSIS) теперь поддерживает SQL Server на Linux, и новый пакет позволяет вам запускать пакеты SSIS в Linux из командной строки.

Новый компонент Горизонтальное увеличение масштаба для SSIS значительно упрощает запуск SSIS на множестве компьютеров.

Источник OData и диспетчер подключений OData теперь поддерживают подключение к веб-каналам OData в Microsoft Dynamics AX Online и Microsoft Dynamics CRM Online.

Службы SQL Server 2017 Master Data Services (MDS).

Обновление с SQL Server 2012, SQL Server 2014 или SQL Server 2016 до SQL Server 2017 Master Data Services обеспечит вам улучшенную производительность и усовершенствованный интерфейс.

Теперь вы можете просматривать на странице «Обозреватель веб-приложения» отсортированный список сущностей, коллекций и иерархий.

Использование хранимой процедуры позволило значительно ускорить промежуточную обработку миллионов записей.

Кроме того, улучшена работа при разворачивании на странице «Управление группами» папки «Сущности» для назначения моделям разрешений. Страница «Управление группами» находится в веб-приложении в разделе «Безопасность».

Дополнительные сведения о назначении разрешений: Назначение разрешения для объекта модели (службы Master Data Services).

Службы SQL Server 2017 Analysis Services (SSAS).

SQL Server Analysis Services 2017 включает множество улучшений для табличных моделей. К ним относятся следующие объекты.

Табличный режим стал параметром установки по умолчанию для Analysis Services.

Безопасность на уровне объектов для защиты метаданных табличных моделей.

Возможность легко создавать связи на основе полей дат.

Новые источники получения данных (Power Query) и поддержка существующих источников данных DirectQuery для запросов на языке M.

Редактор DAX для SSDT.

Подсказки по кодированию — продвинутая функция для оптимизации обновления данных больших табличных моделей в памяти.

Поддержка уровня совместимости 1400 для табличных моделей. Чтобы создать новый проект табличной модели с уровнем совместимости 1400 или перевести уже существующий проект на этот уровень, загрузите и установите SQL Server Data Tools (SSDT) 17.0 RC2.

Современный интерфейс получения данных для табличных моделей с уровнем совместимости 1400.

Свойство «Скрыть – члены» скрывает пустые элементы в неоднородных иерархиях.

Новое действие «Строки детализации» для конечного пользователя, позволяющее показать подробности по статистических данным. Функции SELECTCOLUMNS и DETAILROWS для создания выражений со строками детализации.

DAX-оператор IN для указания множества значений.

Службы SQL Server 2017 Reporting Services (SSRS).

Службы SQL Server Reporting Services больше не доступны для установки с помощью программы установки SQL Server. Перейдите в Центр загрузки Майкрософт, чтобы загрузить службы Microsoft SQL Server 2017 Reporting Services.

В отчетах теперь доступны комментарии, позволяющие сообщать свою точку зрения и взаимодействовать с другими пользователями. Для комментариев также доступны вложения.

В последних выпусках построителя отчетов и SQL Server Data Tools вы можете создавать собственные запросы DAX для поддерживаемых табличных моделей данных SQL Server Analysis Services, перетаскивая нужные поля в конструкторах запросов.

В целях разработки и настройки современных приложений службы SSRS теперь поддерживают API RESTful, полностью совместимый с OpenAPI.

Машинное обучение в SQL Server 2017.

Службы R в SQL Server были переименованы в Службы машинного обучения SQL Server, чтобы отразить появление поддержки Python в дополнение к языку R. Вы можете использовать службы машинного обучения (в базе данных) для запуска сценариев R или Python в SQL Server либо установить сервер машинного обучения Майкрософт (автономный) для развертывания и использования моделей R и Python, не требующих SQL Server.

Разработчикам в SQL Server теперь доступны обширные библиотеки машинного обучения и искусственного интеллекта для Python в экосистеме открытого кода, а также последние инновации от Майкрософта.

revoscalepy — этот эквивалент RevoScaleR (Python) включает параллельные алгоритмы для линейных и логистических регрессий, деревьев решений, усиленных деревьев и случайных лесов, а также обширный набор API для преобразования и перемещения данных, контекстов удаленного вычисления и источников данных.

microsoftml — этот современный пакет алгоритмов и преобразований для машинного обучения с привязками Python включает инструменты работы с глубокими нейронными сетями, быстрыми деревьями и лесами принятия решений, а также оптимизированные алгоритмы линейной и логистической регрессии. Вы также получите предварительно обученные модели на основе моделей ResNet, которые можно использовать для извлечения образов или анализа мнений.

Практическое использование Python с T-SQL — простое развертывание кода Python с помощью хранимой процедуры sp\_execute\_external\_script. Достигните отличной производительности, используя потоковую передачу данных из процессов SQL в процессы Python и параллелизацию кольца MPI.

Python в контекстах вычислений SQL Server — исследователи данных и разработчики могут выполнять код Python удаленно из своей среды разработки для исследования данных и разработки моделей без перемещения данных.

Собственная оценка — функция PREDICT в Transact-SQL, которую можно использовать для выполнения оценки в любом экземпляре SQL Server 2017, даже если язык R не установлен. Требуется только обучить модель с помощью одного из поддерживаемых алгоритмов RevoScaleR и revoscalepy, а затем сохранить модель в новом компактном двоичном формате.

Управление пакетами — T-SQL теперь поддерживает оператор CREATE EXTERNAL LIBRARY, который предоставляет администраторам баз данных больше возможностей для управления с помощью пакетов R. Используйте роли для управления доступом к частным и общим пакетам, хранения пакетов R в базе данных и предоставления пользователям общего доступа к ним.

Повышение производительности — хранимая процедура sp\_execute\_external\_script оптимизирована для поддержки выполнения пакетных операций для данных columnstore.

# **ГЛАВА III. ЭКСПЛУАТАЦИЯ И ЭКОНОМИЧЕСКИЙ РАСЧЕТ.**

# **3.1. Руководство пользователю.**

Система “Help Desk” позволяет сотрудником офиса без каких-либо проблем связаться с технической поддержкой компании. В этом руководстве будет показан путь заявки, от ее открытия до закрытия.

Заходя в форму подачи заявок (рисунок 3.1) пользователь видит основную информацию о компьютере и его пользователе в левой части формы. Справа же, находяться поля для заполнения заявки, которые в последующем отправляются на сервер в таблицу базы данных. После подачи заявки приходит оповещение, в котором пользователь уведомляется об отправке заявки (рисунок 3.2).

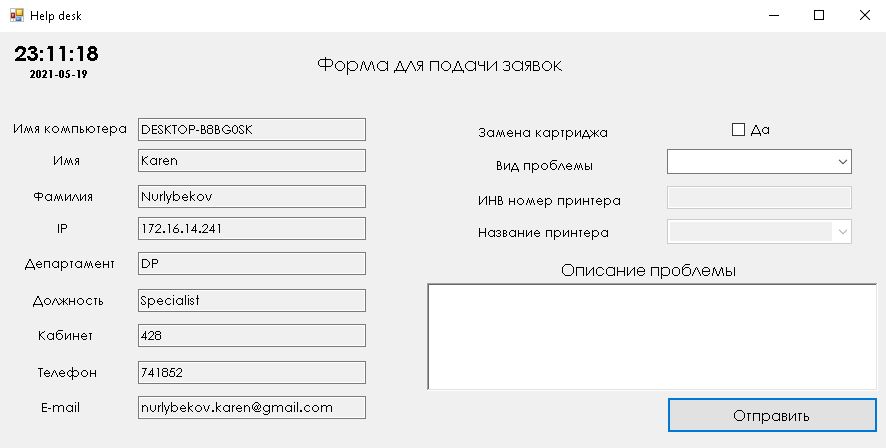


Рисунок 3.1 – Форма подачи заявок.

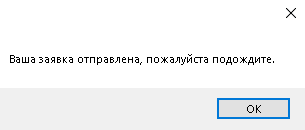


Рисунок 3.2 – Оповещение.

После отправки заявки запись заявки отображается в веб-форме для сотрудников технической поддержки, где отображены все открытые заявки, из которых каждый из сотрудников может свободно выбрать какую заявку будет обрабатывать (рисунок 3.3). На этой веб-форме помимо таблицы с заявками расположены 2 кнопки, которые перенаправляют на другие веб-формы – «Выполненные» и «Оценки», соответственно (рисунок 3.4). Также в каждой строке с заявкой находиться кнопка «Принять» (рисунок 3.5), которая перенаправляет на другую форму, где заполняется акт о выполненных работах.

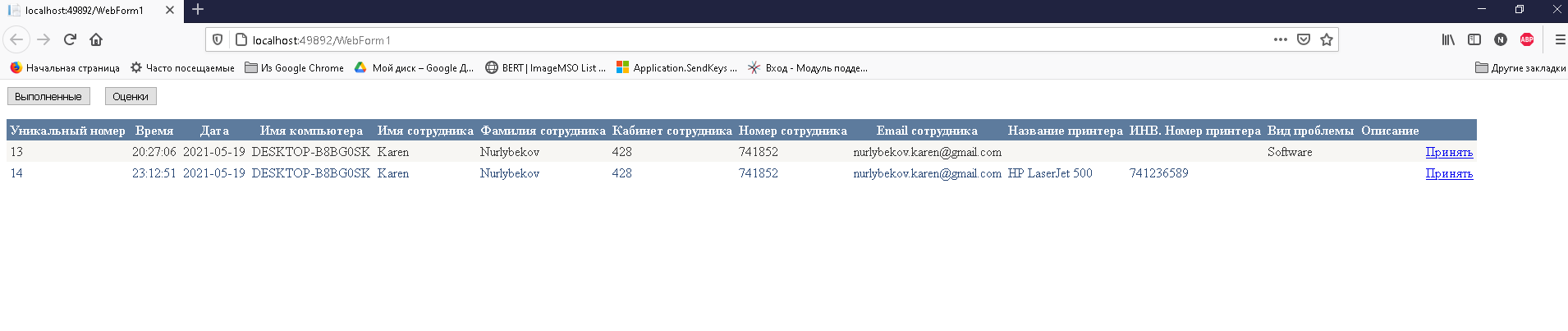


Рисунок 3.3 – Веб-форма для сотрудников технической поддержки.

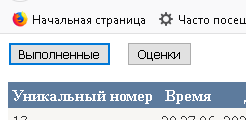


Рисунок 3.4 – Кнопки «Выполненные» и «Оценки».

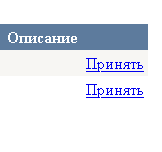


Рисунок 3.5 – Кнопка «Принять».

После нажатия кнопки «Принять» сотрудник переходит к заполнению акта выполненных работ (рисунок 3.6). В данной веб-форме находятся поля для заполнения акта и кнопка, которая позволяет проверить на ту ли заявку заполняется акт (рисунок 3.7). На данный момент кнопка для отправки акта деактивирована, чтобы сотрудник технической поддержки при отправке акта не закрыл несуществующую или активную заявку, и пользователь не получил ложный акт. После отправки акта сотрудником технической поддержки пользователю отправляется уведомление о закрытии заявки (рисунок 3.8), а сама заявка удаляется из таблицы с активными заявками и переноситься в архивные.

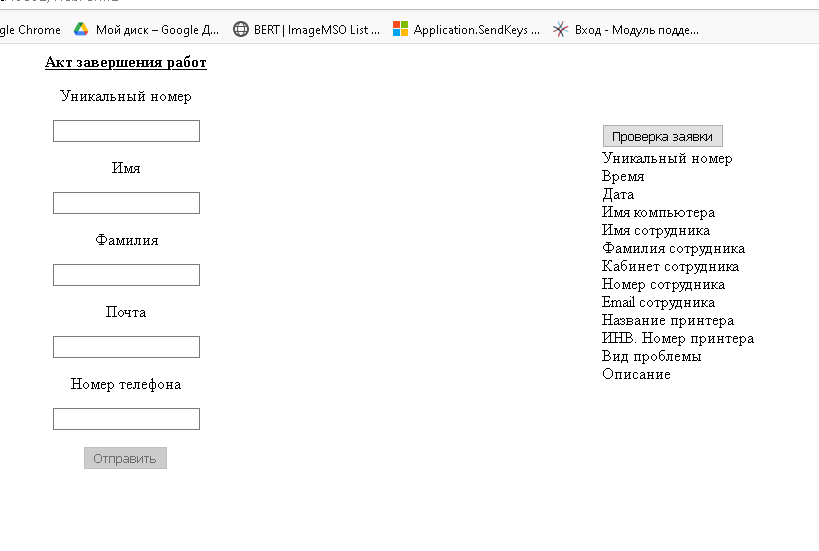


Рисунок 3.6 – Веб-форма «Акт завершения работ».

После проверки заявки кнопка для отправки акта активируется.

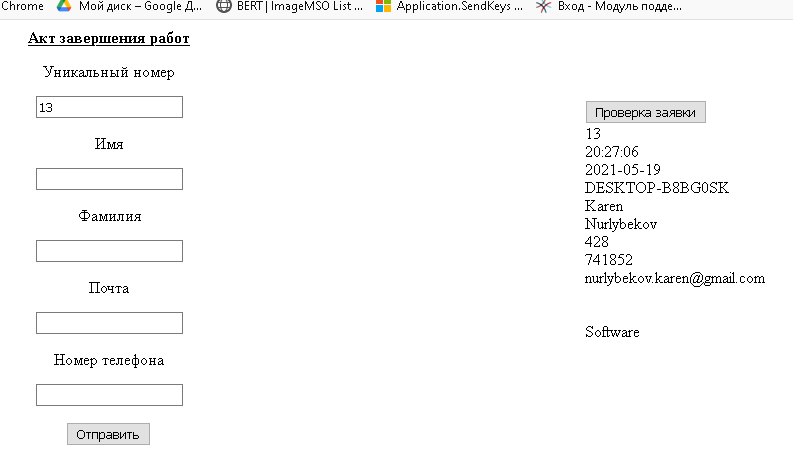


Рисунок 3.7 – Информация о заявке.

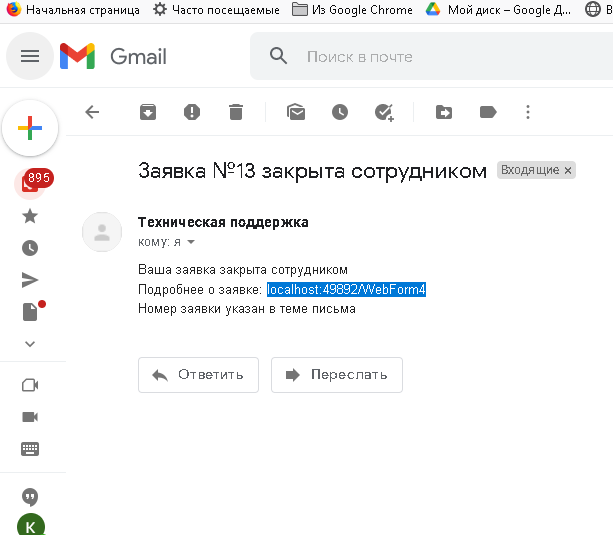


Рисунок 3.8 – Письмо пользователю.

Далее пользователь может пройти по ссылке, указанной в письме и оценить работу сотрудника технической поддержки (рисунок 3.9).

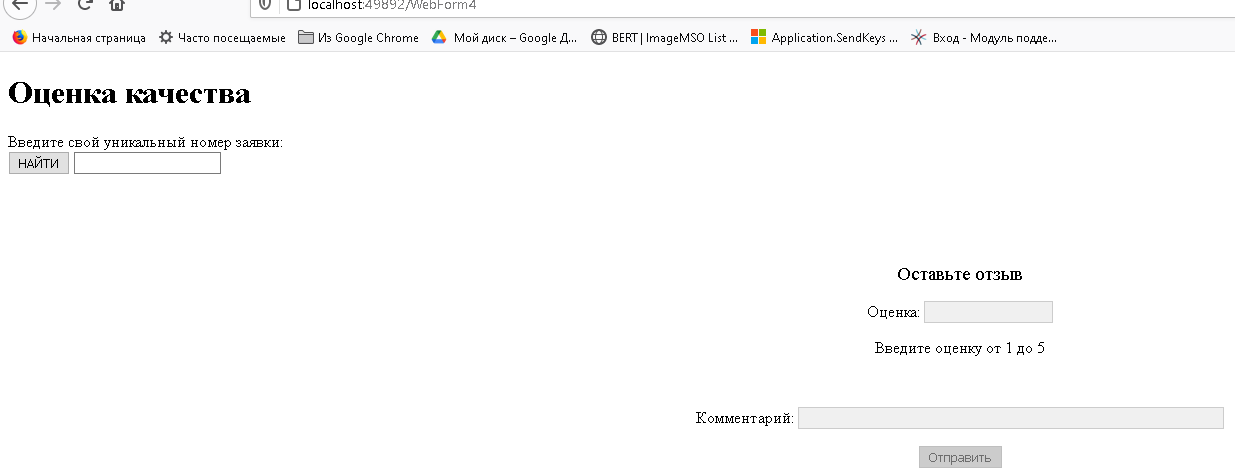


Рисунок 3.9 – Веб-форма «Оценка качества».

В левой верхней части формы можно заметить поле, куда нужно ввести уникальный номер заявки, который содержится в теме письма. Введя уникальный номер заявки и нажав на кнопку принять, то появятся 2 таблицы (рисунок 3.10), в одной из которых будет информация о закрытой заявке, а во второй – о сотруднике, закрывшем заявку.

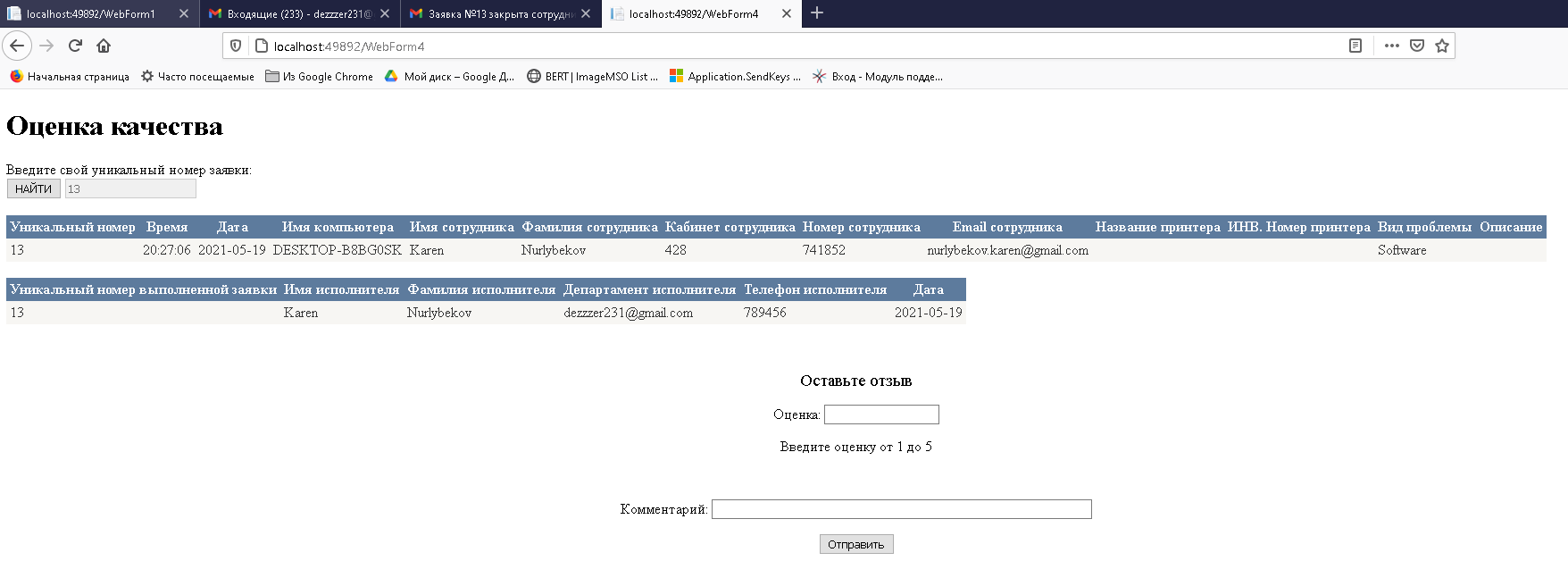


Рисунок 3.10 – Таблицы.

После нажатия кнопки «НАЙТИ» активируются поля для ввода, и кнопка отправить для отправки отзыва и оценки в таблицу базы данных. Эта часть формы была неактивна для того, чтобы пользователь не оставил отзыв и не поставил оценку не той заявке. После отправки отзыва выходит окно, информирующее об отправке отзыва (рисунок 3.11).

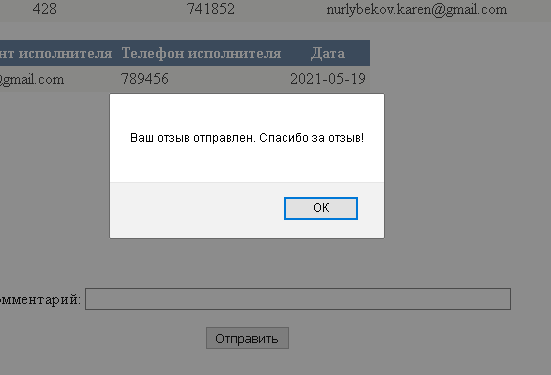


Рисунок 3.11 – Информационное окно.

После того как заявка будет закрыта, информация о ней отправляется в таблицу с архивными заявками (рисунок 3.12). Для просмотра этих заявок надо нажать на кнопку «Выполненные» в Веб-форме с активными заявками.

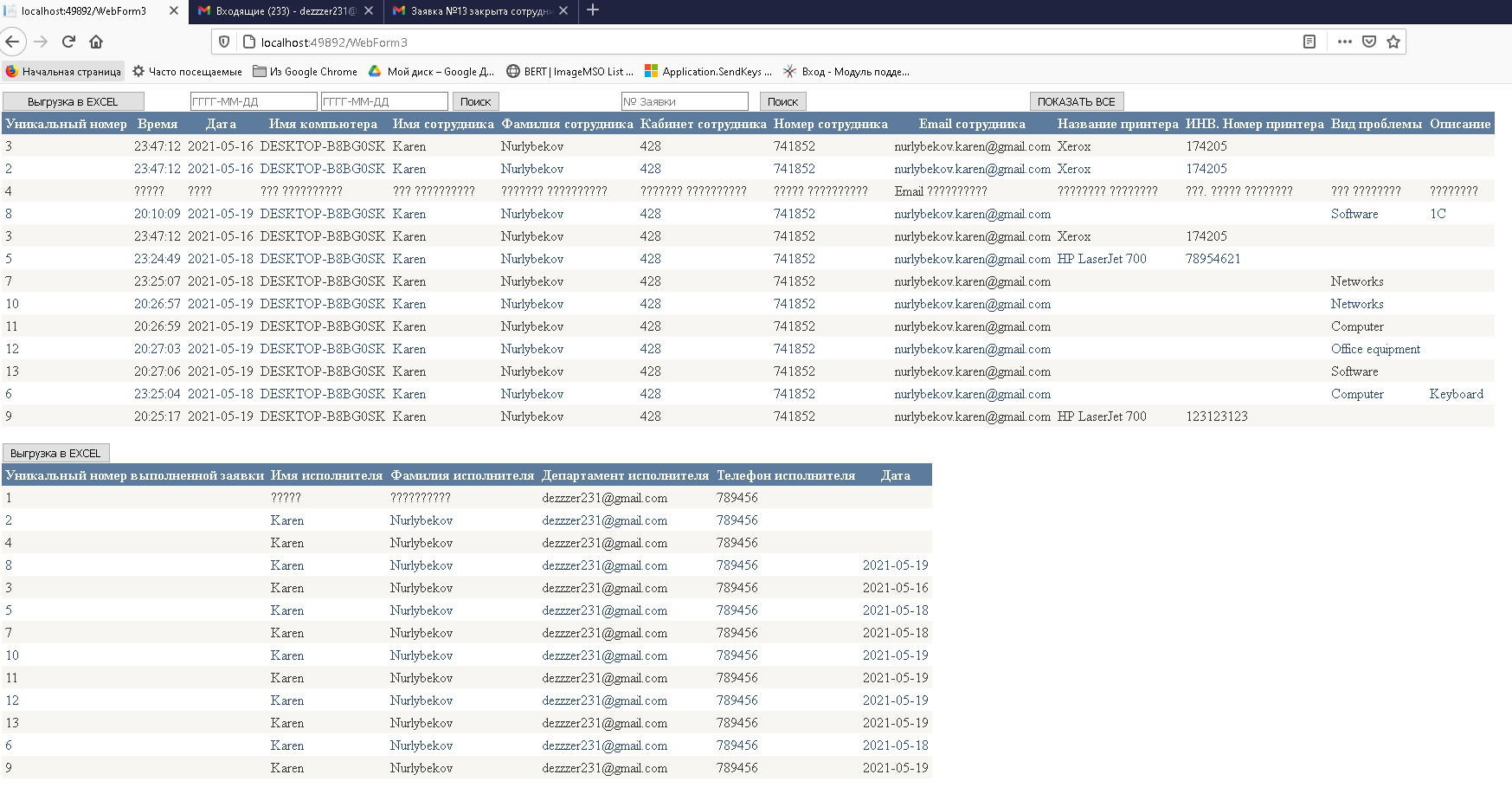


Рисунок 3.12 – Веб-форма «Выполненные».

В верхней части веб-формы находятся кнопка «Выгрузка в EXCEL» (рисунок 3.13), поисковик по датам (рисунок 3.14), поисковик по уникальному номеру заявки (рисунок 3.15) и кнопка «ПОКАЗАТЬ ВСЕ», которая показывает все записи. Рядом со второй таблицей находиться кнопка «Выгрузка в EXCEL». Если заполнить поля для ввода даты, то будет производиться поиск по заданным датам (рисунок 3.16). Поиск по номеру заявки позволяет вывести только одну заявку (рисунок 3.17). Чтобы лишний раз не копировать таблицу, через веб-форму есть выгрузка таблицы в EXCEL файл (рисунок 3.18).



Рисунок 3.13 – Кнопка «Выгрузка в EXCEL».



Рисунок 3.14 – Поисковик по датам



Рисунок 3.15 – Поисковик по уникальному номеру заявки.



Рисунок 3.16 – Поиск по датам.

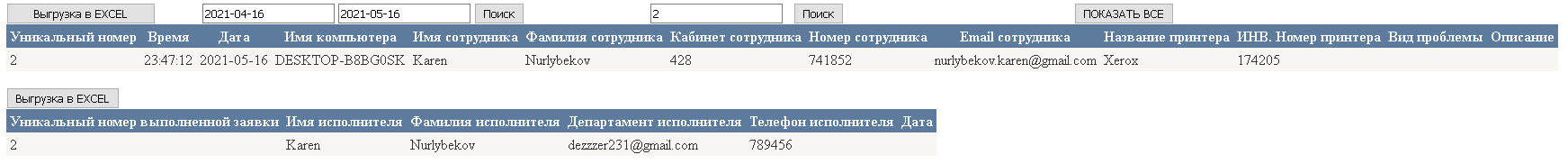


Рисунок 3.17 – Поиск по уникальному номеру заявки.

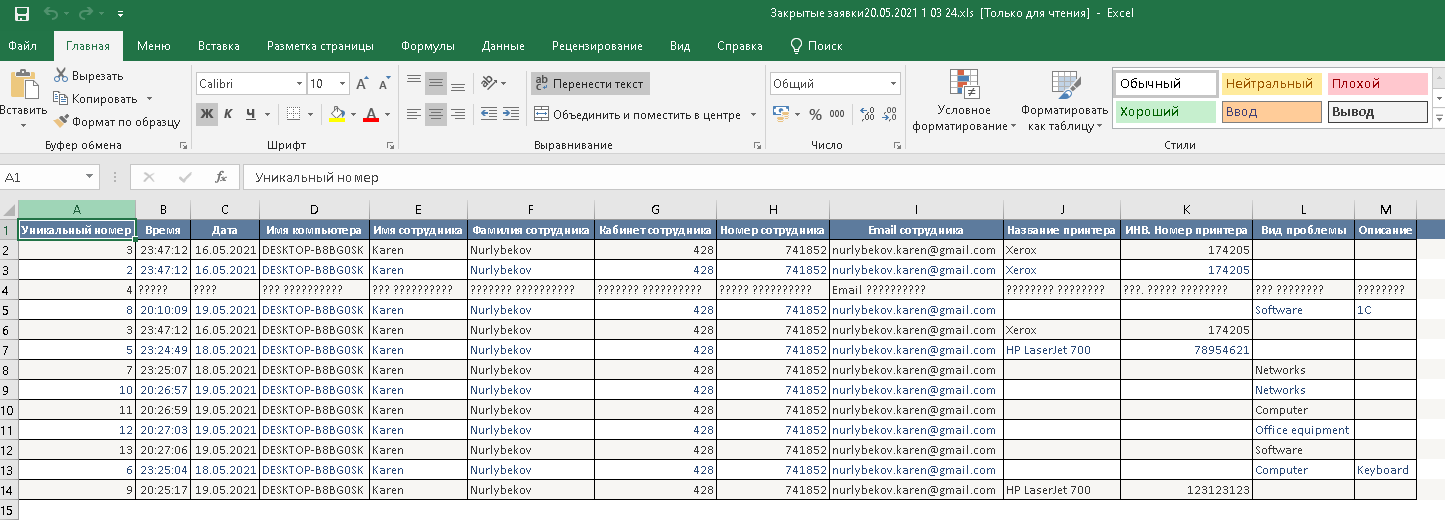


Рисунок 3.18 – Таблица EXCEL.

Начальник отдела, его заместитель, директор могут смотреть отзывы (рисунок 3.19). Для просмотра этих отзывов надо нажать на кнопку «Оценки» в Веб-форме с активными заявками.

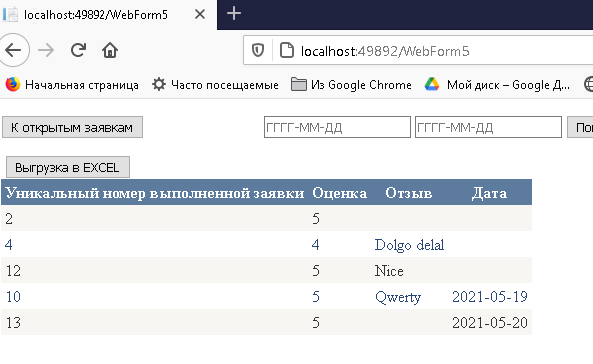


Рисунок 3.19 – Веб-форма «Оценки».

В данной веб-форме также присутствует поиск по дате (рисунок 3.20), выгрузка в EXCEL и кнопка, возвращающая нас в веб-форму с открытыми заявками.

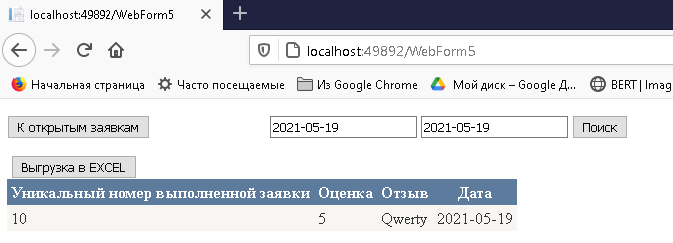


Рисунок 3.20 – Поиск в таблице БД в веб-форме «Оценки».

# **3.2. Расчет затрат на создание системы и ее экономическая эффективность.**

Для создания такой запутанной и сложной системы как Help Desk потребуется много ресурсов.

Перед тем как рассчитать полную стоимость создания системы разработчику нужно ответить на некоторые вопросы как:

* Сколько займет время на создание;
* Сколько форм будет задействовано;
* Какого точная стоимость одной страницы формы;
* Стоимость аренды хостинга;
* Стоимость дизайна системы;
* Стоимость и время на заполнение базы данных;
* Расходы на персонал.

После того, как мы нашли ответы на вышеперечисленные вопросы, то мы уже можем строить таблицу расчетной стоимости проекта (таблица 1).

Таблица 1 – Расчет стоимости.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тип расхода | Время и день | Сумма |
|  | Расходы на интернет | Месяц | 112,1 USD |
|  | Заработная плата разработчика | 20д по 8ч | 1500 USD |
|  | Электричество | 20д по 8ч | 651,2 KZT |
|  | Лицензия MS Visual Studio 2019 Professional | Месяц | 100 USD |

Итог: ~1715 USD

Для создания системы потребуется стабильное и высокоскоростное интернет соединение. Можно воспользоваться услугами от АО «Казахтелеком». У АО «Казахтелеком» имеется комплексное решение для бизнеса.

На создание системы уходит 1,5 месяца. Это считая все выходные дни 30 рабочих дней.

За час своей работы разработчик получает 9,375 USD. Считая тот факт, что время работы за 1 день составляет 8 часов разработчик за день получает 75 USD. За всю работу после окончания разработчик получает 2000 USD.

Создание одной формы стоит ~300 USD. Система состоит из 5 веб-форм и 1 формы. Получается разработчик будет получать зарплату не за время, потраченное на рабочем места, а с каждой формы и БД. Получается:

6\*300 = 1800 USD

2000 – 1800 = 200 USD (Для создания БД)

Для создания системы нам понадобиться виртуальный сервер. Такой сервер для БД нам может предоставить Google Cloud Platform.

SQL Server ценовая политика.

Общая стоимость Cloud SQL для MS SQL состоит из:

1. Цены на процессор и память.
2. Цены на хранение и сети.
3. Лицензирование.

Цена на CPU и память (таблица 2).

Для специализированных экземпляров вы выбираете количество процессоров и объем памяти, который вы хотите, до 96 процессоров и 624 ГБ памяти. Цены на процессоры и память зависят от региона, в котором находится ваш экземпляр.

Для создания было выбрано расположение сервера во Франкфурте.

Таблица 2 – Цены на процессор и память.

|  |  |
| --- | --- |
| Услуга | Цена в месяц |
| vCPUs | $36.18 за 1 vCPU |
| Память | $6.13 за 1 GB |
| HA vCPUs | $72.36 за 1 vCPU |
| HA Память | $12.26 за 1 GB |

Цены на хранилище и сеть (таблица 3).

Стоимость хранилища и сети SQL-сервера зависит от региона, в котором расположен экземпляр. В случае с моей системой расположение экземпляра было выбрано во Франкфурте, поэтому ниже приведены цены для этого региона.

Таблица 3 – Цены на хранение и сети.

|  |  |
| --- | --- |
| Цена | Цена в месяц |
| Хранение | * $0.204 за 1ГБ на SSD диске * $0.096 за 1ГБ на SSD диске для бэкапов |
| HA Хранение(Высокопроизводительное) | * $0.408 за 1ГБ на SSD диске * $0.096 за 1ГБ на SSD диске для бэкапов |

Лицензирование (таблица 4).

Помимо, например, и цен на ресурсы, MS SQL Server также имеет компонент лицензирования, основанный на выбранном издании. Высокая доступность или региональные экземпляры понесут только затраты на единую лицензию для активного ресурса.

Таблица 4 – Цены на хранение и сети.

|  |  |
| --- | --- |
| License | Цена за час |
| Enterprise | $0.47 |
| Standard | $0.13 |
| Web | $0.01134 |

Для лицензирования Microsoft SQL Server требуется, чтобы каждому виртуальному ЦП в экземпляре виртуальной машины была назначена лицензия на ядро ​​с минимум четырьмя ядрами для каждого экземпляра. Для выполнения этих требований с экземпляров с менее чем 4 виртуальными ЦП будет взиматься плата за SQL Server в 4 раза превышающую лицензионную ставку. Для экземпляров с 4 или более виртуальными ЦП вы будете платить за количество лицензий SQL Server, равное количеству виртуальных ЦП.

Складывая все статьи расходов на конфигурацию сервера, то мы можем вывести цену для конфигурации нашего сервера (таблица 5). Мною будет использоваться самая минимальная конфигурация, поскольку для разработки ее вполне хватает, как и для презентации проекта будующим заказчикам системы.

Таблица 5 – Пример минимальной конфигурации для MS SQL.

|  |  |
| --- | --- |
| Детали конфигурации | Цена в месяц |
| 1 CPU  3.75 GB RAM  20 GB SSD  Без места для бэкапов  Стандартная лицензия | $432.31 |

Итоговая цена.

На электричество, программиста и лицензию для MS Visual Studio 2019 Professional будет потрачено ~1715 USD. Суммируя эту сумму с минимальной конфигурацией для MS SQL получается итоговая сумма в размере 2147,31USD.

Варианты заработка на этой системе:

* Подписка.
* Разовое приобретение.

При выборе плана подписки может рассматриваться помесячная и погодовая оплата. К примеру, первый месяц будет бесплатный, а последующие платные. Также при погодовой оплате можно рассмотреть систему скидок, например – если платить помесячно, то получиться так, что компания-заказчик в сумме потратила 350 USD, а если оформить сразу полгодовую оплату, то компания заказчик заплатит 290 USD, то есть на 60 USD меньше, чем при помесячной оплате, и на 5 USD в месяц меньше чем при помесячной оплате.

Рассматривая разовое приобретение лицензии, то куда логичнее выбрать цену, которая будет покрывать всю стоимость разработки системы, но и не настолько большую чтобы вариант с подпиской не был в приоритете у заказчиков. К примеру, сумма в размере 4000 USD будет более чем достаточной и для дохода владельцу компании и для покрытия затрат при создании системы.

Чтобы покрыть всю стоимость разработки системы, то одна компания-заказчик должна иметь оплаченную годовую подписку на 7,5 лет вперед или на 6 лет при помесячной подписке.

# **ЗАКЛЮЧЕНИЕ**

При написании дипломного проекта были рассмотрены цели, задачи, технические спецификации и похожие проекты для создания системы HELPDESK. Целью дипломного проекта было создать данную систему для поддержки пользователей персональных компьютеров, орг. техники, которая будет превосходить по функционалу, цене и удобству в сравнение с другими конкурентами, и она была выполнена.

В ходе написания дипломной работы разработан система «Helpdesk» на языке C#. Сталкивался со многими проблемами при помощи видео уроков и в Интернете нашел некоторые решения ошибок. Также консультировался у дипломного руководителя и у специалистов предприятий, где уже запущенны похожие системы. Для себя вспомнил, что такое работа с классами и переменными, многое узнал о создании дизайна программ, написании кода. На данном этапе могу считать свою работу и программу готовой к использованию. Итоговое приложение было протестировано, и результат проверки дал положительный результат. В настоящее время широко в нашу жизнь внедряются информационные технологии. На сегодняшний день, наверное, не осталось никакой области, где бы они ни применялись. Одной из сфер деятельности, в которой информационные технологии получили широкое распространение.

В результате дипломной работы были выполнены следующие поставленные задачи и цели:

1. Выбрал язык программирования.

Был выбран язык C# так как это объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения. C# относиться к языкам компилируемого типа, поэтому он обладает всеми преимуществами таких языков. Также он объединяет лучшие идеи современных языков программирования Java, C++, Visual Basic и т.д. Из-за большого разнообразия синтаксических конструкций и возможности работать с платформой .Net, C# позволяет быстрее, чем любой другой язык, разрабатывать программные решения. C# отличается надежностью и элегантностью из-за большого разнообразия синтаксических конструкций. Большая надёжность была достигнута из-за работы CLR машины, ведь в отличии от других компилятор CLR запускает разработанное приложение на виртуальном процессоре. Поэтому в случаях возникновения каких-либо ошибок, это никак не повлияет на работу других программ в системе, но это также обозначает, что для запуска программы необходимо дополнительное время. Соответственно программы написанные на языке программирования C# более надёжные, но менее быстрые (нежели те же программы написанные на C++).

1. Выбор IDE.

Важнейшим элементом в процессе разработки приложения является выбор правильной IDE, зависящий не только от платформы, но и уровня собственной подготовки. Давайте познакомимся с наиболее популярными из них методом «от противного», представляя не столько их преимущества, сколько наиболее часто встречаемые укоры со стороны разработчиков.

Для создания системы была использована среда разработки Visual Studio, где мною были использованы WindowsForm C# и WebForm ASP .NET. ASP.NET Web Forms является платформой веб-приложений и одной из нескольких моделей программирования, поддерживаемых технологией Microsoft ASP.NET технологии. Приложения Web Forms могут быть написаны на любом языке программирования, который поддерживает общее время выполнения языка, например, на C# или Visual Basic. Основными строительными блоками страниц Web Forms являются серверные элементы управления, которые являются многоразовыми компонентами, ответственными за визуализацию HTML разметки и реагирование на события.

Интегрированная среда разработки программ Microsoft Visual Studio является уникальным примером активного инновационного развития и воплощает в себе целый сплав новых идей и инструментов в области разработки программного обеспечения. За 19 лет, с 1995 г., выпущено 12 ее версий, каждая из которых содержит большое число нововведений, включая новые языки программирования, средства командной разработки программ и многие другие.

Интегрированная среда Visual Studio активно развивается фирмой Microsoft и является одной из наиболее популярных в мире интегрированных сред разработки. За 19 лет выпущено 12 версий среды Visual Studio, реализованы такие новые языки, как C#, Visual C++ (и его расширение Managed C++ для платформы .NET), Visual Basic и его расширение для платформы .NET, функциональный язык F#, средства управления проектами, средства коллективной разработки и тестирования программ, средства разработки облачных приложений для платформы Microsoft Azure; выпущена и активно развивается многоязыковая объектно-ориентированная платформа Microsoft.NET. Все эти результаты являются уникальными, инновационными, новейшими, имеющими широкое распространение в мире.

Так же хочется отметить Windows Forms — интерфейс программирования приложений (API), отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework. Данный интерфейс упрощает доступ к элементам интерфейса Microsoft Windows за счет создания обёртки для существующего Win32 API в управляемом коде. Причём управляемый код — классы, реализующие API для Windows Forms, не зависят от языка разработки. То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, С++, так и на VB.Net, J# и др. С одной стороны, Windows Forms рассматривается как замена более старой и сложной библиотеке MFC, изначально написанной на языке C++. С другой стороны, WF не предлагает парадигмы, сравнимой с MVC. Для исправления этой ситуации и реализации данной функциональности в WF существуют сторонние библиотеки. Одной из наиболее используемых подобных библиотек является User Interface Process Application Block, выпущенная специальной группой Microsoft, занимающейся примерами и рекомендациями, для бесплатного скачивания. Эта библиотека также содержит исходный код и обучающие примеры для ускорения обучения.

1. Интуитивный интерфейс.

Для дизайна любого интерфейса одним из главных свойств является понятность и доступность использования. Любому пользователю должно быть легко взаимодействовать с системой. Если человеку будет непонятно или трудно разбираться с приложением, он, скорее всего, просто откажется от этого. Следовательно, любой веб-сайт или приложение должны быть понятными для пользователя на уровне интуиции.

Загромождение интерфейса значками, переходами и прочими управляющими элементами не приведут ни к чему хорошему. Доступность в данном случае будет только кажущейся, а на деле заставит пользователя долго разбираться, что для чего существует и куда оно приведет. Дипломной работе постарался сделать все, чтобы создать интерфейсы системы понятными, но с минимальной загруженностью. Любой пользователь должен максимально быстро разобраться в программе.

1. Отчетность.

В системе доступны данные в виде таблиц EXCEL за выбранный период. Инструмент отлично подходит для общей оценки деятельности Хелпдеска и анализа нагрузки. Данные отчеты позволяют показать начальству работу технической поддержки, удовлетворенность пользователей оказываемыми услугами, проблемами с аппаратурой, офисной техникой. В последствии это положительно скажется на финансовом планировании IT департамента.

1. Оптимизация кода.

Все элемента кода задействованы в программах, все лишнее убрано. Как бонус код был сделан максимально читаемым, чтобы заказчики, при желании и отдельную плату, могли его дорабатывать.

1. Взаимодействие с пользователями.

Пользователи системы имеет возможность оставить оценку и отзыв об закрытой заявке. Данное решение поможет повысить уровень качества оказываемых услуг, узнать если сотрудник технической поддержки сделал свою работу недобросовестно или же эти оценки могут стать основанием для премирования сотрудников технической поддержки, что тоже выведет качество оказываемых услуг на неведомо для этого высокий уровень.

Данная система однозначно найдет своего потребителя так как системы такого типа как HELP DESK так как они заметно снижают время, которое сотрудники компаний тратят на технические проблемы и все что с ними связанное.

Преимущества Help desk для работы компаний:

* Помогает заявкам быстро поступать в обработку
* Система предоставляет возможность руководству контролировать работу менеджеров компании и качество их обслуживания, а также эффективность работы ИТ-отдела
* Облегчает работу инфраструктуры, которая в крупных организациях бывает очень сложной

Лично я считаю свою работу полностью завершенной, а цель – создание системы “HelpDesk” успешно исполненной. Как и все существующие системы, моя имеет пространство для развития. Одно из направлений – это учетные записи сотрудников, которые являются полезной функцией, но от них можно я отказаться на начальном этапе, хотя в последующем их можно добавить. Также хочется отметить, что можно было сделать возможным отправлять заявку не только через форму, установленную на локальном компьютере, но и через браузер, тем самым создав систему, которая будет полностью располагаться на сервере. Более того, улучшенную версию систему можно будет продавать дороже, или просто ее продавать, а нынешний вариант использовать как бесплатную версию.

# **Список использованной литературы.**

* 1. <https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio>
  2. <https://metanit.com/sharp/net/8.1.php>
  3. <https://stackoverflow.com/questions/15196381/how-can-i-display-a-messagebox-in-asp-net>
  4. <https://habr.com/ru/post/479394/>
  5. <https://visualstudio.microsoft.com/ru/vs/>
  6. <https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019>
  7. <https://cloud.google.com/sql/pricing>
  8. <https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:HelpDesk>
  9. <https://cyberleninka.ru/article/n/migratsiya-baz-dannyh-mysql-v-oblachnuyu-sredu-google-cloud-sql-i-vozmozhnosti-eyo-primeneniya-v-sfere-obrazovaniya>
  10. <https://helpdeskeddy.ru/help-desk-chto-eto-i-zachem-nuzno>
  11. https://habr.com/ru/company/deskun/blog/331354/
  12. https://helpdeskeddy.ru/vozmoznosti/ocenka-urovnja-servisa
  13. http://tehpis.ru/info/articles/clear\_interface/
  14. https://mysmartservice.com/blog/chto-takoe-helpdesk
  15. https://softline.ru/uploads/32/9c/65/c9/8c/f6/43/2a/12/origin.pdf
  16. The C# Programming Yellow Book: Learn to program in C# from first principles (Rob Miles)
  17. Работа с базами данных Microsoft SQL Server (А. И. Митин)
  18. Google Cloud Platform in Action (JJ Geewax)
  19. SQL Server 2017: A Practical Guide for Beginners (Gregory Blake)
  20. Building & Managing a World Class IT Help Desk (Bob Wooten)

# **Приложение А.**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.Sql;

using System.Data.SqlClient;

namespace WindowsFormsApp7

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox1.Checked == true)

{

comboBox1.Enabled = false;

textBox9.Enabled = true;

comboBox2.Enabled = true;

}

else

{

comboBox1.Enabled = true;

textBox9.Enabled = false;

comboBox2.Enabled = false;

}

}

private void Form1\_Load(object sender, EventArgs e)

{

timer1.Start();

label16.Text = DateTime.Now.ToLongTimeString();

textBox9.Enabled = false;

comboBox2.Enabled = false;

string computer = System.Environment.MachineName;

SqlConnection conn = new SqlConnection("Data Source=34.89.216.108;Initial Catalog=hd;Persist Security Info=True;User ID=sqlserver;Password=@dmin2010");

conn.Open();

SqlCommand com = conn.CreateCommand();

com.CommandText = @"SELECT \* FROM workers";

SqlDataReader prov = com.ExecuteReader();

while (prov.Read())

{

textBox10.Text = prov["compname"].ToString();

}

prov.Close();

if (textBox10.Text == computer)

{

SqlCommand comm = conn.CreateCommand();

comm.CommandText = @"SELECT \* FROM workers WHERE compname= '" + textBox10.Text + "'";

SqlDataReader read = com.ExecuteReader();

while (read.Read())

{

textBox1.Text = read["name"].ToString();

textBox2.Text = read["surname"].ToString();

textBox3.Text = read["ip"].ToString();

textBox4.Text = read["dep"].ToString();

textBox5.Text = read["pos"].ToString();

textBox6.Text = read["room"].ToString();

textBox7.Text = read["phonenumber"].ToString();

textBox8.Text = read["email"].ToString();

}

read.Close();

}

else

{

MessageBox.Show("ERROR");

}

prov.Close();

conn.Close();

}

private void button2\_Click(object sender, EventArgs e)

{

SqlConnection conn = new SqlConnection("Data Source=34.89.216.108;Initial Catalog=hd;Persist Security Info=True;User ID=sqlserver;Password=@dmin2010");

conn.Open();

SqlCommand com = conn.CreateCommand();

if (checkBox1.Checked == true)

{

com.CommandText = "INSERT INTO dbo.probll(time, date, compname, sname, ssurname, sroom, snumber, semail, nomprint, nazvprint, opisanie) VALUES('" + label16.Text + "', '" + label17.Text + "','" + textBox10.Text + "', '" + textBox1.Text + "','" + textBox2.Text + "', '" + textBox6.Text + "','" + textBox7.Text + "', '" + textBox8.Text + "','" + textBox9.Text + "', '" + comboBox2.Text + "','" + richTextBox1.Text + "')";

com.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Ваша заявка отправлена, пожалуйста подождите.");

textBox9.Clear();

comboBox2.SelectedIndex = -1;

richTextBox1.Clear(); DateTime time; time = DateTime.Now;

}

else

{

com.CommandText = "INSERT INTO dbo.probll(time, date, compname, sname, ssurname, sroom, snumber, semail, vidprob, opisanie) VALUES('" + label16.Text + "', '" + label17.Text + "', '" + textBox10.Text + "', '" + textBox1.Text + "','" + textBox2.Text + "', '" + textBox6.Text + "','" + textBox7.Text + "', '" + textBox8.Text + "','" + comboBox1.Text + "','" + richTextBox1.Text + "')";

com.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Ваша заявка отправлена, пожалуйста подождите.");

comboBox1.SelectedIndex = -1;

richTextBox1.Clear();

}

}

private void Timer1\_Tick(object sender, EventArgs e)

{

label16.Text = DateTime.Now.ToLongTimeString();

string theDate = dateTimePicker1.Value.ToString("yyyy-MM-dd");

label17.Text = theDate;

timer1.Start();

}

}

}

WebForm1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

using System.IO;

namespace WebApplication3

{

public partial class WebForm1 : System.Web.UI.Page

{

string connectionString = @"Data Source=34.89.216.108;Initial Catalog=hd;Persist Security Info=True;User ID=sqlserver;Password=@dmin2010";

protected void Page\_Load(object sender, EventArgs e)

{

using (SqlConnection sqlCon = new SqlConnection(connectionString))

{

sqlCon.Open();

SqlDataAdapter sqlDa = new SqlDataAdapter("SELECT \* FROM probll", sqlCon);

DataTable dtbl = new DataTable();

sqlDa.Fill(dtbl);

GridView1.DataSource = dtbl;

GridView1.DataBind();

}

}

protected void lnkSelect\_Click(object sender, EventArgs e)

{

Response.Redirect("~/WebForm2.aspx");

}

protected void Button1\_Click(object sender, EventArgs e)

{

Response.Redirect("~/WebForm3.aspx");

}

protected void Button2\_Click(object sender, EventArgs e)

{

Response.Redirect("~/WebForm5.aspx");

}

}

}

WebForm2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

using System.Net.Mail;

namespace WebApplication3

{

public partial class WebForm2 : System.Web.UI.Page

{

SqlConnection conn = new SqlConnection("Data Source=34.89.216.108;Initial Catalog=hd;Persist Security Info=True;User ID=sqlserver;Password=@dmin2010");

public void Button2\_Click(object sender, EventArgs e)

{

}

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

conn.Open();

SqlCommand com = conn.CreateCommand();

SqlCommand cmd = conn.CreateCommand();

SqlCommand cmt = conn.CreateCommand();

com.CommandType = CommandType.Text;

cmd.CommandType = CommandType.Text;

cmt.CommandType = CommandType.Text;

com.CommandText = "INSERT INTO dbo.done(numb, progname, progsurn, progdep, progphone, date) VALUES('" + textBox5.Text + "', '" + TextBox1.Text + "','" + TextBox2.Text + "', '" + TextBox3.Text + "', '" + TextBox4.Text + "', '" + Label8.Text + "')";

com.ExecuteNonQuery();

cmd.CommandText = "DELETE FROM dbo.probll WHERE num= '" + textBox5.Text + "'";

cmd.ExecuteNonQuery();

cmt.CommandText = "INSERT INTO dbo.rezz(num, time, date, compname, sname, ssurname, sroom, snumber, semail, nazvprint, nomprint, vidprob, opisanie) VALUES('" + textBox5.Text + "', '" + Label7.Text + "', '" + Label8.Text + "', '" + Label9.Text + "','" + Label10.Text + "', '" + Label11.Text + "','" + Label12.Text + "', '" + Label13.Text + "','" + Label14.Text + "','" + Label15.Text + "','" + Label16.Text + "', '" + Label17.Text + "','" + Label18.Text + "')";

cmt.ExecuteNonQuery();

TextBox1.Text = "";

TextBox2.Text = "";

TextBox3.Text = "";

TextBox4.Text = "";

textBox5.Text = "";

MailAddress from = new MailAddress("dezzzer231@gmail.com", "Техническая поддержка");

MailAddress to = new MailAddress("nurlybekov.karen@gmail.com");

MailMessage msg = new MailMessage(from, to);

msg.Subject = "Заявка №" + Label6.Text + " закрыта сотрудником";

msg.Body = "Ваша заявка закрыта сотрудником <br>Подробнее о заявке: localhost:49892/WebForm4 <br>Номер заявки указан в теме письма";

msg.IsBodyHtml = true;

SmtpClient smtp = new SmtpClient("smtp.gmail.com", 587);

smtp.EnableSsl = true;

smtp.Credentials = new System.Net.NetworkCredential("dezzzer231@gmail.com", "lollol952505dj");

smtp.Send(msg);

conn.Close();

Response.Redirect("~/WebForm1.aspx");

}

protected void Button2\_Click1(object sender, EventArgs e)

{

using (SqlConnection sqlCon = new SqlConnection("Data Source=34.89.216.108;Initial Catalog=hd;Persist Security Info=True;User ID=sqlserver;Password=@dmin2010"))

{

sqlCon.Open();

SqlCommand com = sqlCon.CreateCommand();

com.CommandText = @"SELECT \* FROM dbo.probll WHERE num= '" + textBox5.Text + "'";

SqlDataReader read = com.ExecuteReader();

while (read.Read())

{

Label6.Text = read["num"].ToString();

Label7.Text = read["time"].ToString();

Label8.Text = read["date"].ToString();

Label9.Text = read["compname"].ToString();

Label10.Text = read["sname"].ToString();

Label11.Text = read["ssurname"].ToString();

Label12.Text = read["sroom"].ToString();

Label13.Text = read["snumber"].ToString();

Label14.Text = read["semail"].ToString();

Label15.Text = read["nazvprint"].ToString();

Label16.Text = read["nomprint"].ToString();

Label17.Text = read["vidprob"].ToString();

Label18.Text = read["opisanie"].ToString();

}

read.Close();

sqlCon.Close();

}

Button1.Enabled = true;

}

}

}

1. https://helpdeskeddy.ru/help-desk-chto-eto-i-zachem-nuzno [↑](#footnote-ref-1)
2. https://mysmartservice.com/blog/chto-takoe-helpdesk [↑](#footnote-ref-2)
3. https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:HelpDesk [↑](#footnote-ref-3)
4. https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio [↑](#footnote-ref-4)
5. https://cyberleninka.ru/article/n/migratsiya-baz-dannyh-mysql-v-oblachnuyu-sredu-google-cloud-sql-i-vozmozhnosti-eyo-primeneniya-v-sfere-obrazovaniya [↑](#footnote-ref-5)