СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc152804328)

[1 ЦЕЛЬ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ 5](#_Toc152804329)

[1.1 Основные требования по технике безопасности во время прохождения учебной практики 6](#_Toc152804330)

[1.2 Нормативно-правовые документы 6](#_Toc152804331)

[2 ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ 8](#_Toc152804332)

[2.1 Описание предметной области 8](#_Toc152804333)

[2.2 Группы пользователей информационной системы 9](#_Toc152804334)

[2.3 Основные требования, предъявляемые к информационной системе 9](#_Toc152804335)

[3 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ “БЮРО ТЕХНИЧЕСКОЙ ИНВЕНТАРИЗАЦИИ” 11](#_Toc152804336)

[3.1 Моделирование некоторых динамических аспектов системы 11](#_Toc152804337)

[3.1.1 Создание диаграммы вариантов использования 12](#_Toc152804338)

[3.1.2 Создание диаграммы деятельности 13](#_Toc152804339)

[3.2 Разработка функциональной структуры информационной системы. Карта навигации по информационной системе 13](#_Toc152804340)

[3.3 Разработка макета информационной системы “Бюро технической инвентаризации” 14](#_Toc152804341)

[4 РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ “БЮРО ТЕХНИЧЕСКОЙ ИНВЕНТАРИЗАЦИИ” 20](#_Toc152804342)

[4.1 Моделирование и разработка базы данных 20](#_Toc152804343)

[4.2 Разработка программных модулей информационной системы “Бюро технической инвентаризации” 24](#_Toc152804344)

[5 РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ 54](#_Toc152804345)

[5.1 Назначение эксперимента. Выбор и обоснование методики проведения тестирования 54](#_Toc152804346)

[5.2 Технология тестирования 54](#_Toc152804347)

[5.3 Результаты проведения тестирования 55](#_Toc152804348)

[ЗАКЛЮЧЕНИЕ 56](#_Toc152804349)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 57](#_Toc152804350)

# ВВЕДЕНИЕ

Бюро технической инвентаризации (БТИ) — это специализированные организации, которые ведут учет состояния всех объектов недвижимости, включая индивидуальные и многоквартирные дома, квартиры, дачи и земельные участки. Они собирают сведения об основных технических характеристиках строений, их расположении, планировке, площади, а также о всех изменениях, которые внесены за время их эксплуатации.

Бюро технической документации оформляет широкий набор документов, которые могут потребоваться при оформлении различных сделок с недвижимостью или согласовании работ, связанных с перепланировкой.

# ЦЕЛЬ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ

Цели разработки информационной системы для бюро технической инвентаризации могут включать в себя:

1. Улучшение эффективности и оперативности процессов обработки заявок на создание и выдачу документов клиентами.

2. Обеспечение точной и своевременной информации о выполненных заявках.

3. Оптимизация процессов взаимодействия с клиентами для более удобного процесса оформления документов.

4. Повышение уровня обслуживания клиентов через автоматизацию процессов подачи заявок и их оплаты.

Задачи разработки информационной системы для бюро технической инвентаризации могут включать в себя:

1. Анализ потребностей клиентов и определение функциональных требований к информационной системе.

2. Разработка программного обеспечения для управления заявками.

3. Обеспечение безопасности данных и защиты от несанкционированного доступа к информации о заявках, клиентах.

4. Обучение персонала и поддержка пользователей при внедрении новой информационной системы.

Разработка информационной системы для бюро технической инвентаризации должна быть направлена на повышение эффективности работы с заявками на получение документов.

Для достижения целей и задач будет разработана и спроектирована информационная система “Бюро технической инвентаризации”, нацеленная на оптимизацию процесса внесения и хранения данных в этой предметной области, в рамках учебной практики по модулю ПМ 01 (Разработка модулей программного обеспечения для компьютерных систем).

## 1.1 Основные требования по технике безопасности во время прохождения учебной практики

К работе в компьютерном классе допускаются лица, ознакомленные с данной инструкцией по технике безопасности и правилам поведения. Необходимо неукоснительно соблюдать правила по технике безопасности, т.к. нарушение этих правил может привести к поражению электрическим током, вызвать возгорание и навредить вашему здоровью.

* Работа обучающихся в компьютерном классе разрешается только в присутствии преподавателя (лаборанта).
* Во время занятий посторонние лица могут находиться в классе только с разрешения преподавателя.
* Во время перемен между уроками проводится обязательное проветривание компьютерного кабинета с обязательным выходом обучающихся из класса.

Каждый обучающийся в ответе за состояние своего рабочего места и сохранность размещенного на нем оборудования.

Обучающийся, допустивший нарушение по технике безопасности и правилам поведения в компьютерном классе, может быть привлечен к дисциплинарной ответственности. Если нарушение техники безопасности связано с причинением имущественного ущерба, обучающийся несет и материальную ответственность в установленном законом порядке.

## 1.2 Нормативно-правовые документы

Оформление и содержание технического задания должно соответствовать требованиям стандарта «ГОСТ 19.201-78. ЕСПД. Общие положения.

Техническое задание оформляют в соответствии с ГОСТ 19.106-78 на листах формата 11 и 12 по ГОСТ 2.301-68, как правило, без заполнения полей листа. Номера листов (страниц) проставляются в верхней части листа над текстом.

Лист утверждения и титульный лист оформляют в соответствии с ГОСТ 19.104-78.

Информационную часть (аннотацию и содержание), лист регистрации изменений допускается в документ не включать.

Для внесения изменений или дополнений в техническое задание на последующих стадиях разработки программы или программного изделия выпускают дополнение к нему.

Согласование и утверждение дополнения к техническому заданию проводят в том же порядке, который установлен для технического задания.

Техническое задание должно содержать следующие разделы:

* введение;
* основания для разработки;
* назначение разработки;
* требования к программе или программному изделию;
* требования к программной документации;
* технико-экономические показатели;
* стадии и этапы разработки;
* порядок контроля и приемки;
* в техническое задание допускается включать приложения.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

Вывод по главе: были описаны, техника безопасности во время прохождения учебной практики; нормативно-правовые документы, которые отвечают за стандарты разработки; цели и задачи учебной практики, включая вводную часть, где описана тематика отрасли и её тенденции развития.

# 2 ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ

Предметная область информационной системы — это материальная система или система, характеризующая элементы материального мира, информация о которой хранится и обрабатывается. Предметная область рассматривается как некоторая совокупность реальных объектов и связей между ними.

## 2.1 Описание предметной области

Необходимо разработать и спроектировать информационную систему для эффективного управления процессами подачи и просмотра заявок на получение документов клиентами Бюро технической инвентаризации.

Бюро технической инвентаризации принимает заявки на изготовление нескольких видов документов. Технический паспорт объекта **-** документ, который нужен для сделок с недвижимостью, для споров с налоговой, если некорректно начисляют налог, и если нужно согласовать перепланировку.Справка БТИ — это выписка из техпаспорта с кратким описанием квартиры и дома.Кадастровый паспорт -В кадастровом паспорте на квартиру указывали кадастровый номер, адрес, характеристики и план расположения помещения на этаже. На земельный участок — информацию о кадастровом номере, стоимости, площади, прилагали ситуационный план участка.Справка обинвентаризационной стоимости **-** может понадобиться для оформления наследства, отчуждения или приватизации недвижимости. По инвентаризационной стоимости начисляют налог, и в случае споров с налоговой потребуется такая справка.Справка о соответствии адреса **-** может понадобиться при изменении адреса для внесения уточняющих сведений в документы о праве собственности.

Информационная система "Бюро технической инвентаризации" призвана содействовать повышению эффективности и удобства в процессах подачи и обработки поданных заявок.

## 2.2 Группы пользователей информационной системы

Сотрудники, в зависимости от своих функциональных обязанностей, могут выполнять действия, направленные на мониторинг, управление, анализ данных в рамках проектируемой информационной системы "Бюро технической инвентаризации".

Роли и их Обязанности:

* Администратор: имеет все права управления информационной системы такие как удаление, запись, создание и изменение. Выполняет поставленные задачи для корректной работы информационной системы.
* Начальник: имеет возможность просмотра всех таблиц и добавления информации в таблицы “Техники”, “Исполнители”, “Отделы”.
* Бухгалтер: может просматривать информацию из таблиц “Клиенты”, “Заявки”, “Выполненные заявки” и заполнять эти таблицы.
* Исполнитель: имеет возможность просмотра таблицы “Заявки”.
* Техник: имеет возможность просмотра таблицы “Заявки”.
* Клиент: имеет возможность подать заявку и просмотра таблицы “Выполненные заявки”.

## 2.3 Основные требования, предъявляемые к информационной системе

Система должна позволять авторизовываться/регистрироваться, вести учет заявок сотрудниками и просмотр заявок клиентами. Бухгалтер должен вносить клиентов и заполнять заявки, после передавать начальнику. Исполнитель должен просматривать заявки и вносить правки в заявку, переданную назначенным техником. Техник должен просматривать заявки производить замеры объекта и передавать данные исполнителю. Начальник должен иметь возможность просматривать заявок, клиентов и возможность назначать исполнителей и техников на выполнение заявки. Клиент должен иметь возможность просмотра выполненных заявок. Администратор корректирует информационную систему по запросам Начальника.

Вывод по главе: было описано, что такое предметная область и её описание; группы пользователей, которые будут пользоваться информационной системой и основные требования к ней

# 3 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ “БЮРО ТЕХНИЧЕСКОЙ ИНВЕНТАРИЗАЦИИ”

Проектированием информационных систем называется многоступенчатый процесс их создания и/или модернизации путём применения упорядоченной совокупности методологий и инструментария. Проектирование (в отличие от моделирования) предполагает работу с пока несуществующим объектом и направлено на создание информационной системы в области:

* обработки объектов будущей базы данных,
* написания программ (в том числе – отчётных и экранных форм), обеспечивающих выполнение запросов к данным,
* выполнения учёта функционирования конкретной среды (технологии).

Если выделять стадию проектирования информационных систем в качестве отдельного этапа, то его можно разместить между этапами анализа и разработки. Однако на практике чёткое разделение на этапы, как правило, затруднено или невозможно, поскольку проектирование, формально начинаясь с определения цели проекта, часто продолжается на стадиях тестирования и реализации.

## 3.1 Моделирование некоторых динамических аспектов системы

Диаграммы деятельности применяются для моделирования динамических аспектов поведения системы. Эти динамические аспекты могут включать деятельность на любом уровне абстракции в любом представлении системной архитектуры, включая классы (в том числе активные), интерфейсы, компоненты и узлы.

Использовать диаграммы деятельности для моделирования некоторых динамических аспектов системы вы можете в контексте почти любого моделируемого элемента. Однако чаще вы будете прибегать к таким диаграммам в контексте всей системы, подсистемы, операции или класса. Диаграмму деятельности можно присоединить к варианту использования, чтобы моделировать сценарий, и к кооперации, чтобы моделировать динамические аспекты поведения совокупности объектов.

При разработке программного обеспечения заблаговременное планирование и моделирование значительно упрощают программирование. На этапе создания концептуальной модели для описания функционального назначения используется диаграмма вариантов использования, а для описания логики процедур и бизнес-процессов используется диаграмма деятельности.

### 3.1.1 Создание диаграммы вариантов использования

В приложении 1, на рисунке 1 представлен пример диаграммы вариантов использования информационной системы для бюро технической инвентаризации.

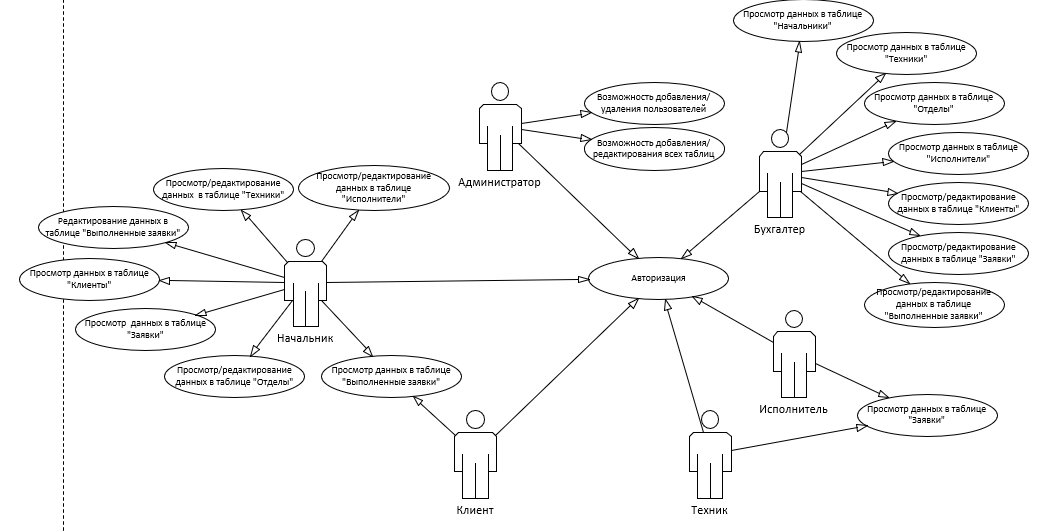


Рисунок 1 - Диаграмма вариантов использования

### 3.1.2 Создание диаграммы деятельности

В приложении 2, на рисунке 2 представлена диаграмма деятельности информационной системы «Бюро технической инвентаризации».



Рисунок 2 - Диаграмма деятельности

## 3.2 Разработка функциональной структуры информационной системы. Карта навигации по информационной системе

Карта навигации по информационной системе «Бюро технической инвентаризации». представлена в приложении 3 на рисунке 3.

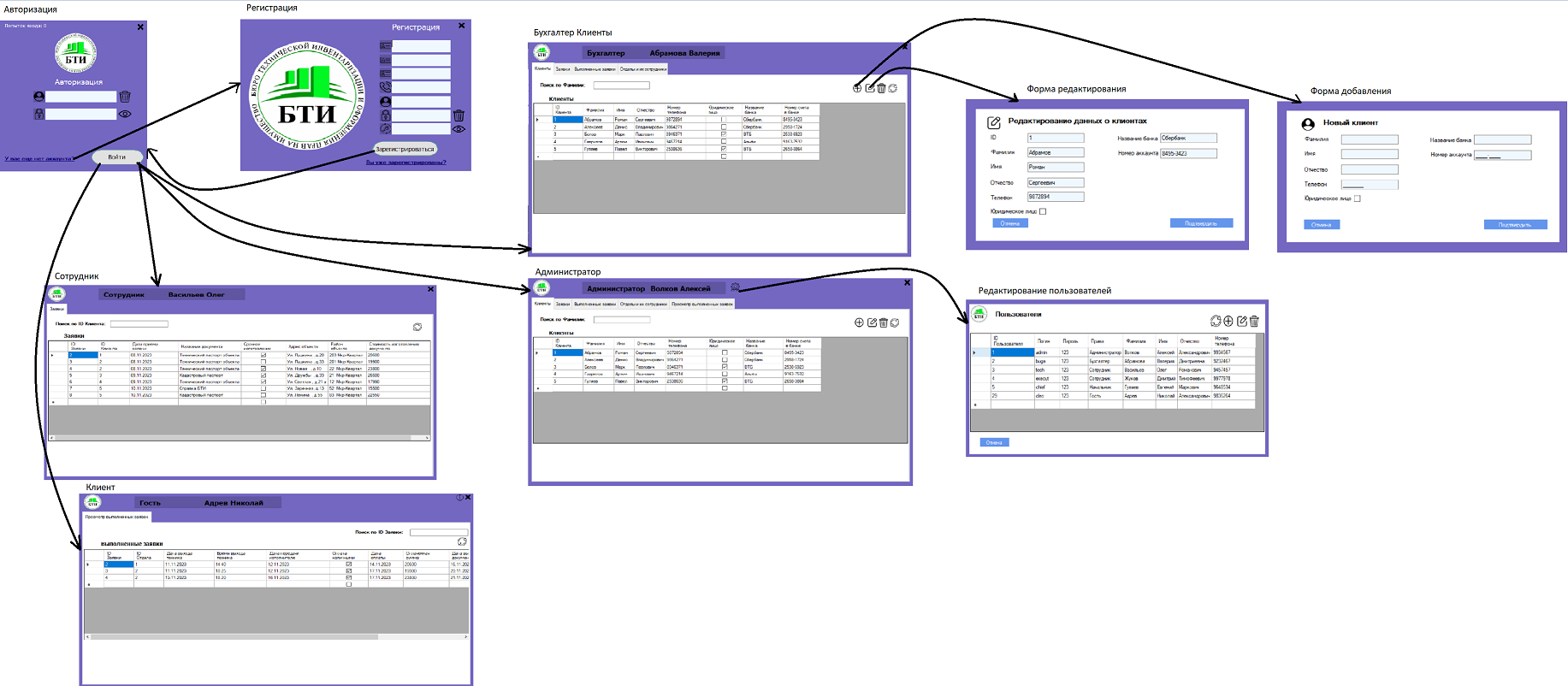


Рисунок 3 – Карта переходов

## 3.3 Разработка макета информационной системы “Бюро технической инвентаризации”

В приложении 4 на рисунке 4 представлено окно авторизации.

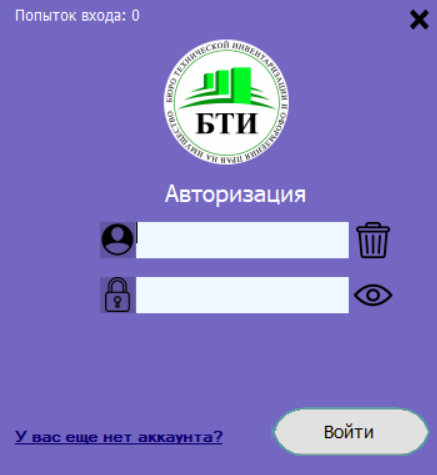


Рисунок 4 – Форма “Авторизация”

На этапе авторизации пользователь может ввести логин и пароль, чтобы продолжить работу в информационной системе, а также показать, скрыть пароль при необходимости, очистить поля или перейти на форму регистрации, если он не зарегистрирован в ИС.

Если пользователь еще не зарегистрирован в системе, он может сделать это, нажав кнопку «У вас еще нет аккаунта?». В этом случае откроется окно регистрации, представленное в приложении 5 на рисунке 5.

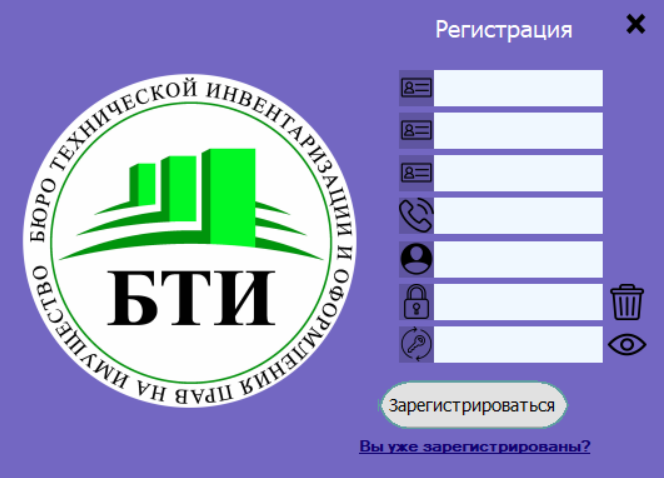


Рисунок 5 – Форма “Регистрация”

На этапе регистрации пользователю необходимо придумать свой логин и пароль, а также ввести ФИО и номер телефона.

Если пользователь с введенными данными не зарегистрирован в базе данных, в качестве админа, бухгалтера, руководителя филиала или же изготовителя и продавца, то пользователь будет зарегистрирован под правами гостя, так как данная информационная система не даст возможность просмотра той или иной информации без нужных прав доступа. Так же при необходимости пользователь с правами админа может выдать права доступа необходимые для работы с информацией.

После того, как пользователь прошел авторизацию, он может продолжить работу с таблицами на главной форме. В приложении 5 на рисунке 6 – 12 представлены формы, которые будут отображены, в зависимости от прав пользователя в системе.

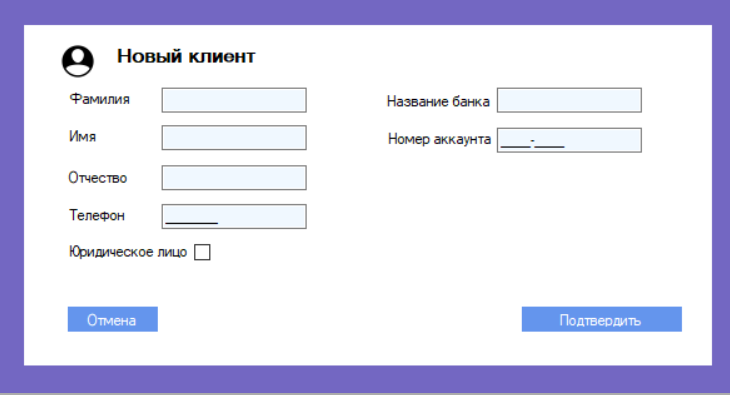


Рисунок 6 – Форма добавления нового клиента

Для добавления новой записи пользователю необходимо заполнить нужные поля и нажать на кнопку «Подтвердить», аналогичным образом выполнены формы добавления данных для других таблицы.

Выбрав конкретную запись и нажав на кнопку редактирования, пользователь переходит на форму, представленную в приложении 5 на рисунке 7.

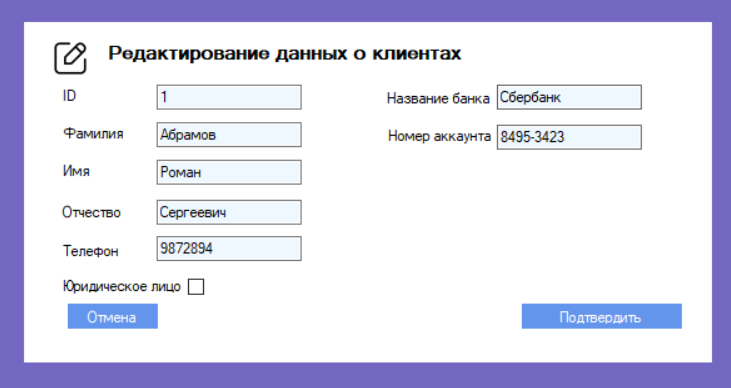


Рисунок 7 – Форма редактирования клиентов

На этой форме будут отображаться данные конкретной записи, которую пользователь выбрал для редактирования, аналогичным образом выполнены формы редактирования данных для других таблицы.

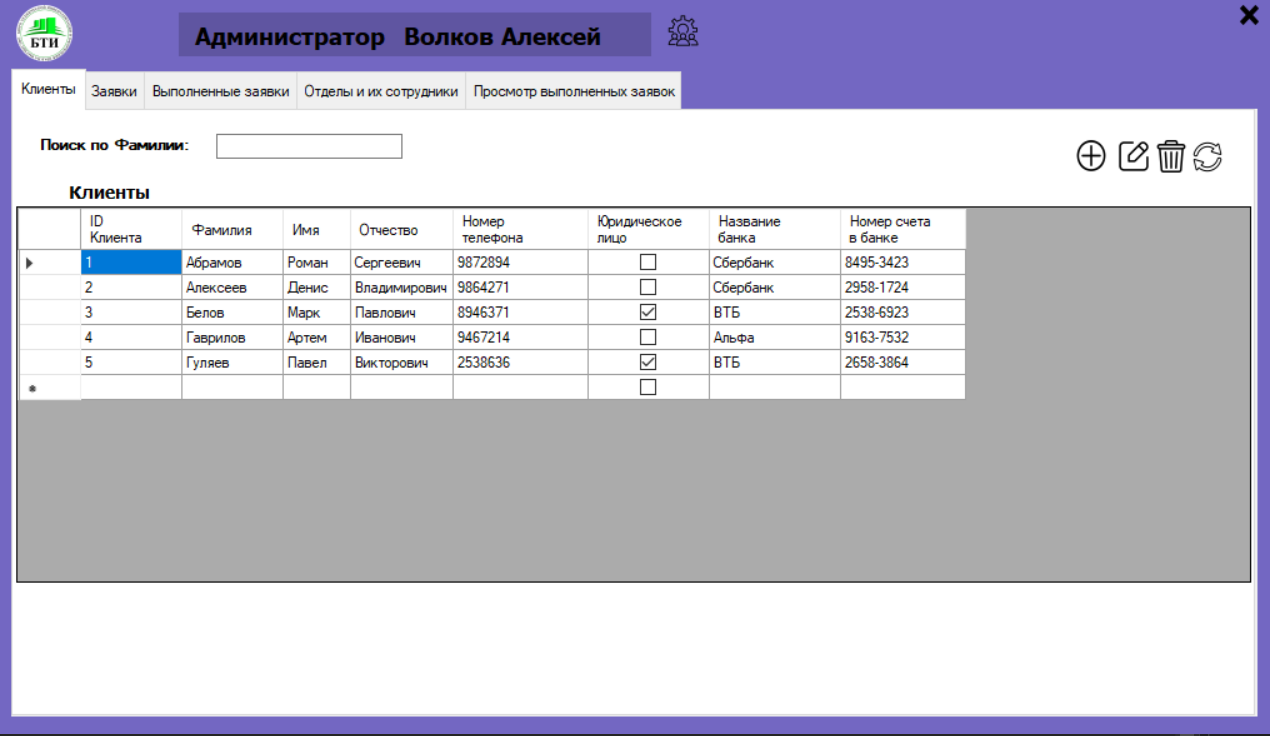


Рисунок 8 – Форма Администратора

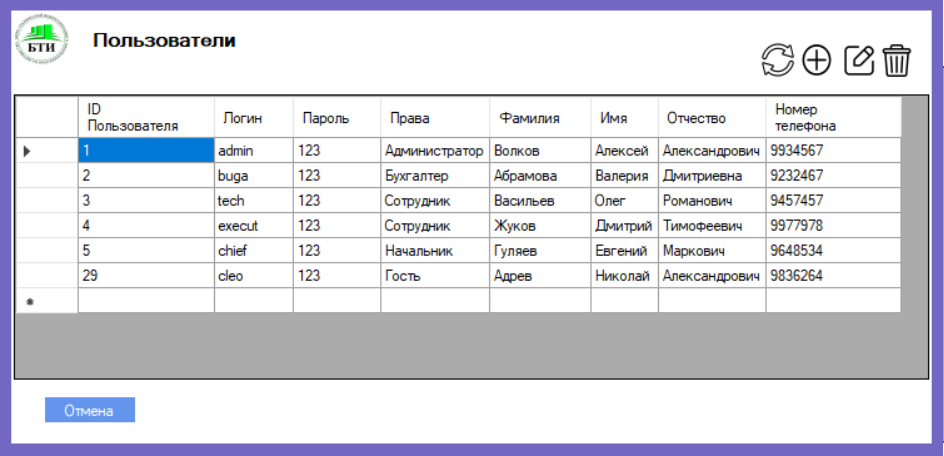


Рисунок 9 – Форма Администратора “Редактирование пользователей”

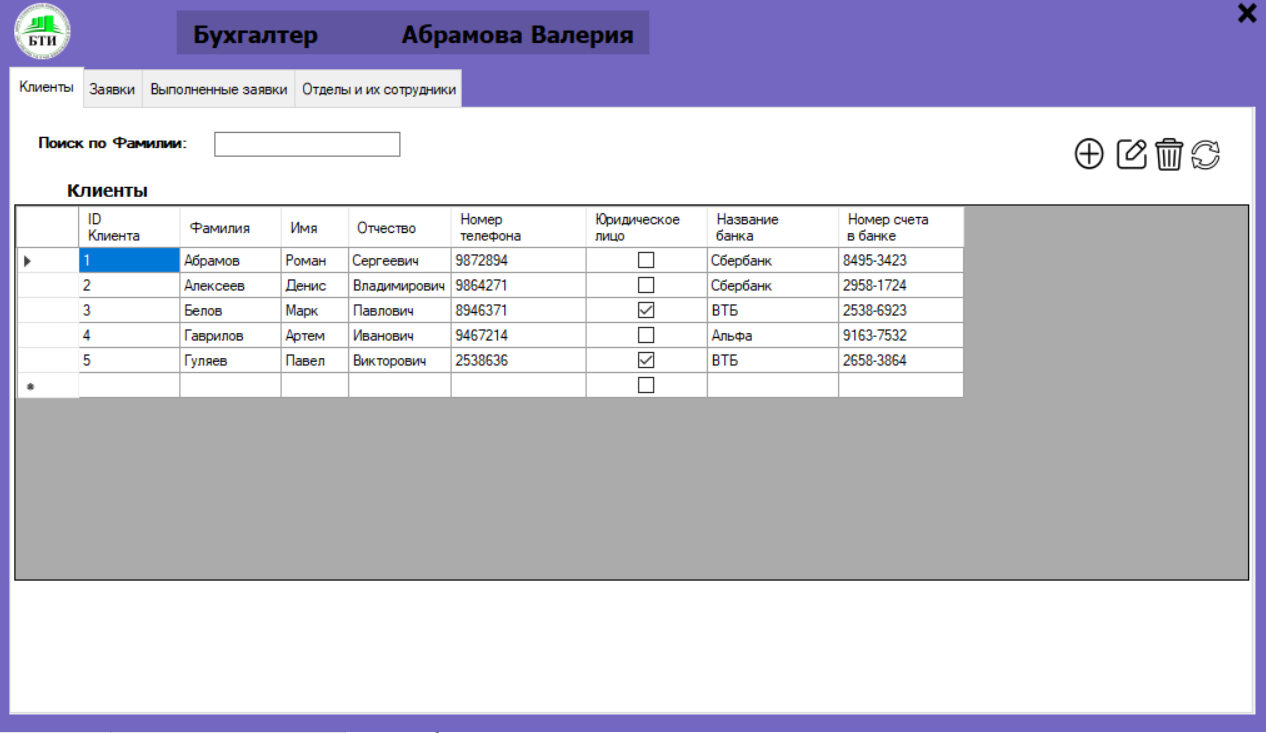


Рисунок 10– Форма Бухгалтера

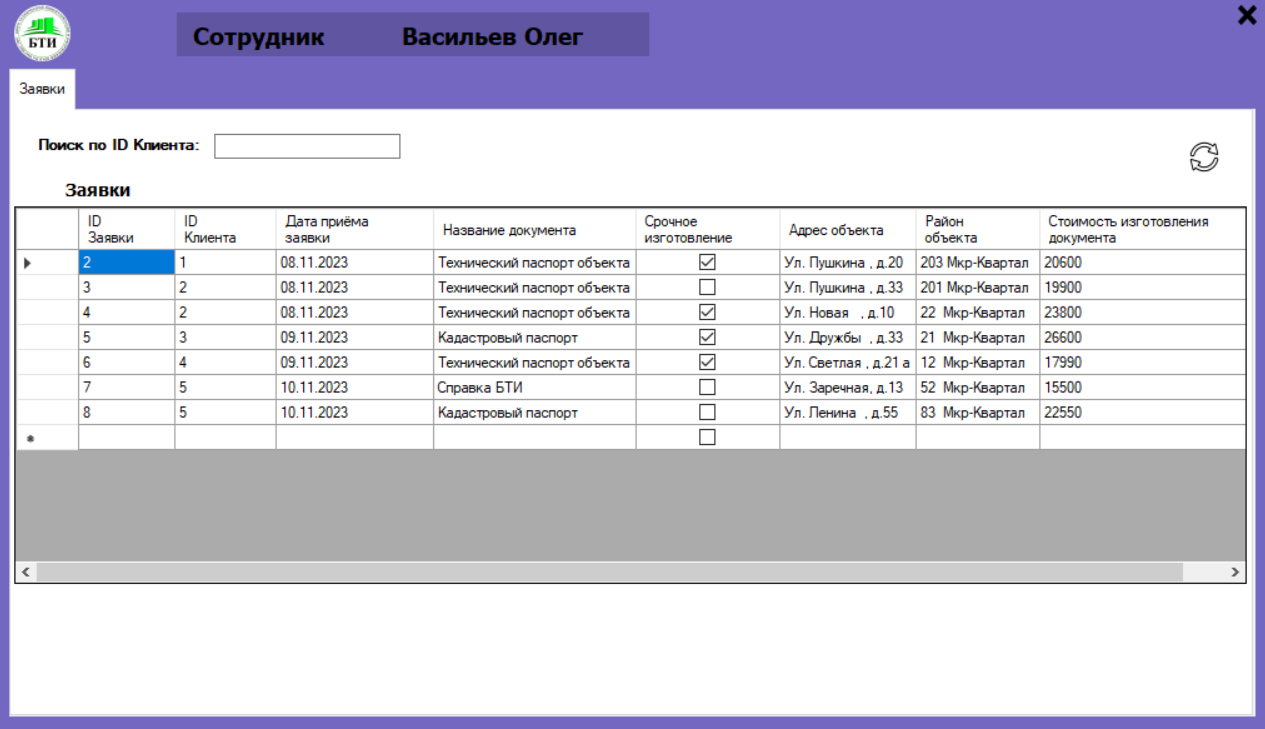


Рисунок 11 – Форма Сотрудника

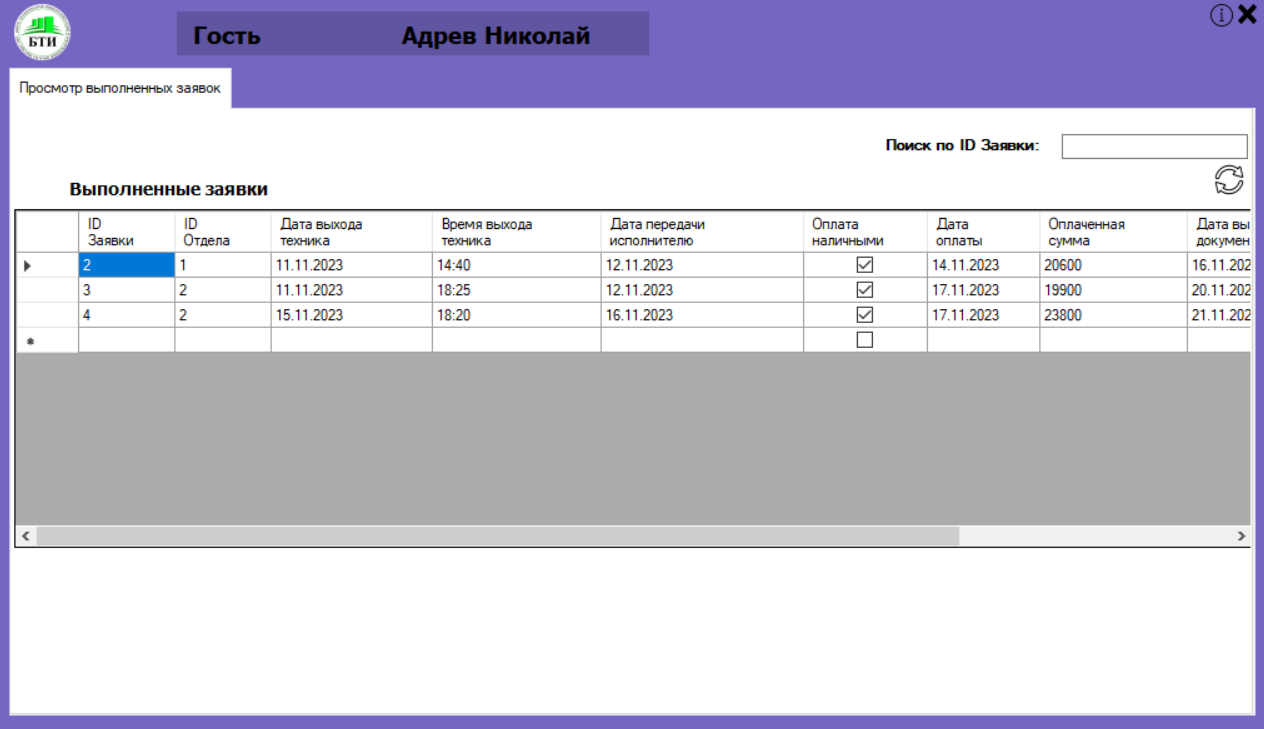


Рисунок 12 – Форма Клиента

В зависимости от роли пользователя осуществляется переход на форму, дозволенную правами, на которой расположены все необходимые таблицы. Например, при авторизации как гость будут отображаться кнопка обновления, строка поиска данных в таблице, так как данной группе пользователю доступен только просмотр данных.

Дизайн выполнен в доступном стиле для того, чтобы пользователь с любым уровнем владения компьютером мог без препятствий работать с программным обеспечением.

Вывод по главе: была спроектирована диаграмма вариантов использования, диаграмма деятельности и макеты будущей информационной системы.

# 4 РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ “БЮРО ТЕХНИЧЕСКОЙ ИНВЕНТАРИЗАЦИИ”

Прототипирование — это один из этапов разработки, который заключается в продумывании содержания и расположения важных элементов интерфейса. Прототип — это моделирование конечного продукта. Это интерактивный макет, который может иметь любую степень точности. Основная цель создания прототипов – проверить, насколько последователен путь пользователя, и выявить препятствия, которые могут возникнуть в процессе его взаимодействия с продуктом.

Прототипы не только позволяют проверить удобство разрабатываемого продукта до начала написания кода, они также приводят к неожиданным открытиям и новым идеям, которые могут вывести проектируемый продукт на новый уровень.

## 4.1 Моделирование и разработка базы данных

Исходя из анализа предметной области, можно выделить семь сущностей: «Customers», «Applications», «Departments», «CompletedApplications», «HeadsOfDepartments». «Executors», «TechnicalWorkers».

В приложении 6 на рисунке 13 представлена логическая модель данных базы данных «Бюро технической инвентаризации».

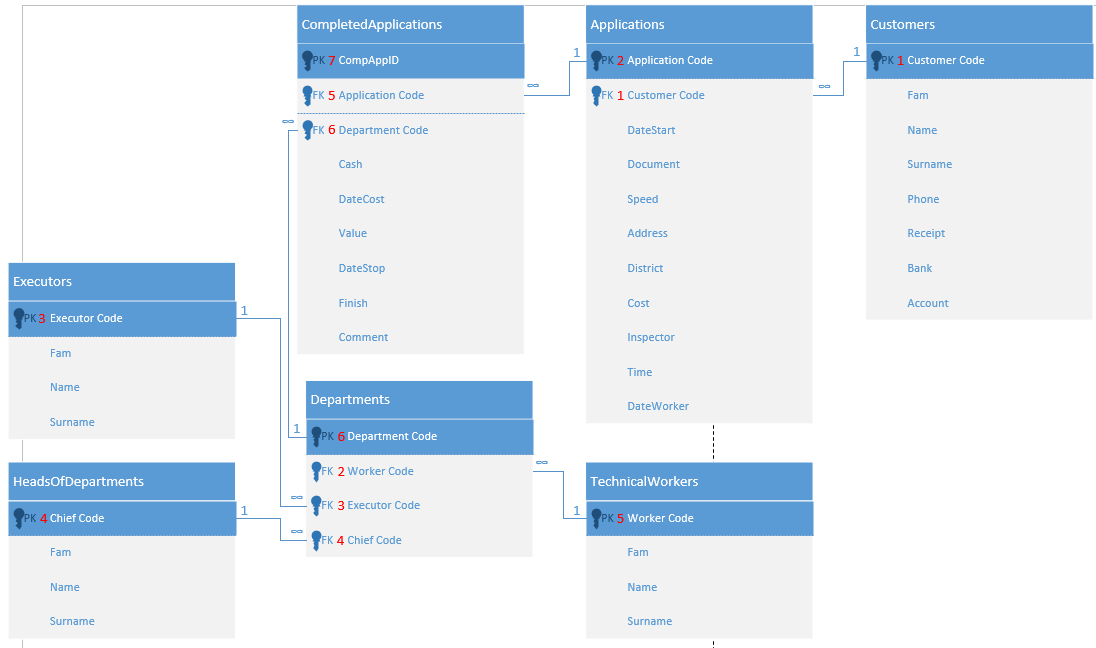


Рисунок 13 – Логическая модель данных

Создание базы данных происходит в MS SQL Server Management Studio, в примере представлено создание базы данных в MS SQL Server.

С помощью раздела «Создание» и конструктора таблиц MS SQL созданы все таблицы и поля, представленные на логической модели данных.

Названия полей и их типы данных представлены в таблицах 1-7.

Таблица 1 – Поля таблицы «Customers»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Customer\_Code | Числовой |
| Fam | Текстовый |
| Name | Текстовый |
| Surname | Текстовый |
| Phone | Текстовый |
| Receipt | Логический |
| Bank | Текстовый |
| Account | Текстовый |

Таблица 2 – Поля таблицы «Applications»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Application\_Code | Числовой |
| Customer\_Code | Числовой |
| DateStart | Дата |
| Document | Текстовый |
| Speed | Логический |
| Address | Текстовой |
| District | Текстовый |
| Cost | Числовой |

Таблица 3 – Поля таблицы «Departments»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Department\_Code | Числовой |
| Chief\_Code | Числовой |
| Executor\_Code | Числовой |
| Worker\_Code | Числовой |

Таблица 4 – Поля таблицы «CompletedApplications»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| CompAppID | Числовой |
| Application\_Code | Числовой |
| Department\_Code | Числовой |
| Inspector | Дата |
| Time | Текстовый |
| DateWorker | Дата |
| Cash | Логический |
| DateCost | Дата |
| Value | Числовой |
| DateStop | Дата |
| Finish | Логический |
| Comment | Тестовый |

Таблица 5 – Поля таблицы «HeadsOfDepartments»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Chief\_Code | Числовой |
| FamName | Текстовый |
| Name | Текстовый |
| Surname | Текстовый |

Таблица 6 – Поля таблицы «Executors»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Executor\_Code | Числовой |
| FamName | Текстовый |
| Name | Текстовый |
| Surname | Текстовый |

Таблица 7 – Поля таблицы «TechnicalWorkers»

| Название поля | Тип данных |
| --- | --- |
| Worker\_Code | Числовой |
| Fam | Текстовый |
| Name | Текстовый |
| Surname | Текстовый |

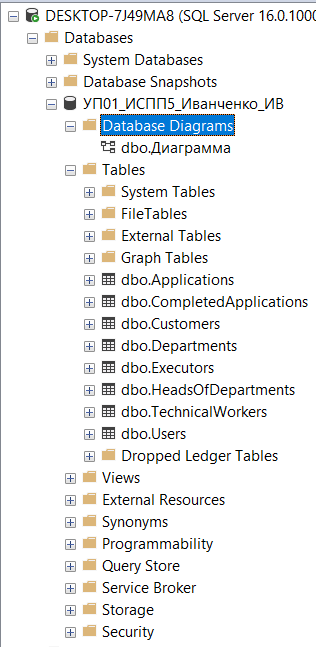


Рисунок 14 – Созданные таблицы

После того, как таблицы созданы, необходимо создать между ними связи.

В приложении 6 На рисунке 12 представлена физическая модель данных, отражающая все таблицы и их поля, а также связи между таблицами.

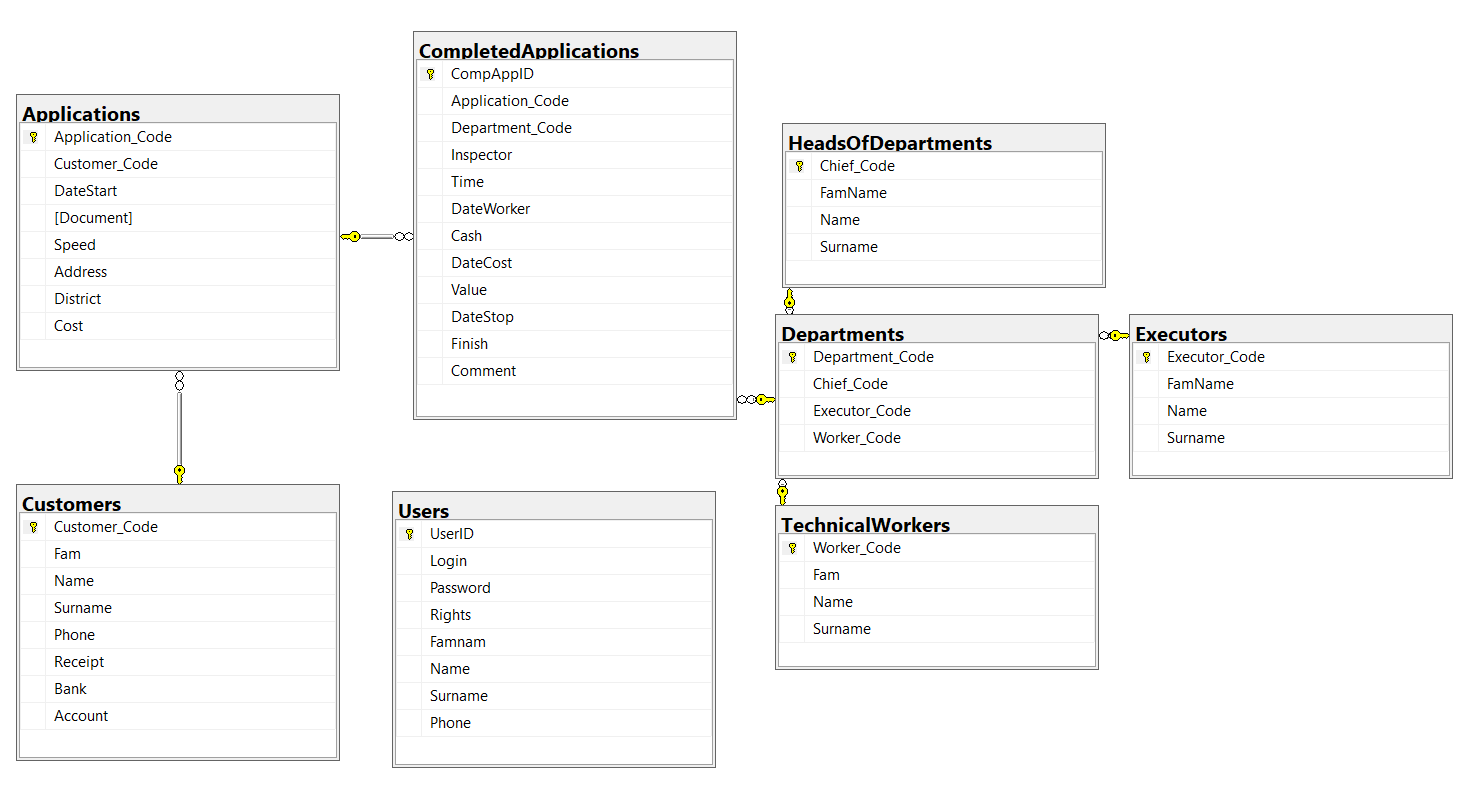


Рисунок 15 – Физическая модель базы данных

## 4.2 Разработка программных модулей информационной системы “Бюро технической инвентаризации”

Разработанные в MS Visual Studio формы продемонстрированы в приложении 7 на рисунках 16-20.

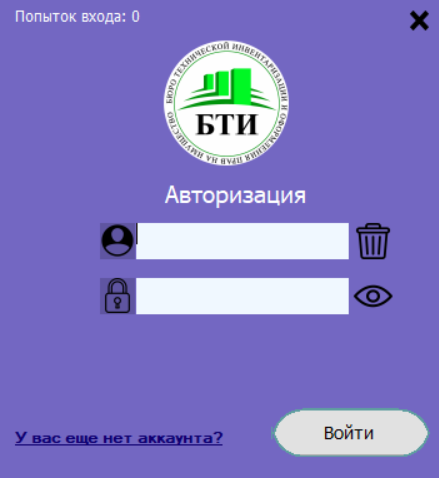


Рисунок 16 – Форма “Авторизация”

Свойства элементов формы Авторизация представлены в приложении 7 в таблице 6.

Таблица 8 – Свойства элементов формы Авторизация

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Авторизация | FormBorderStyle | None |
| Text | Бюро технической инвентаризации |
| StartPosition | CenterScreen |
| Size | 470; 472 |
| ShowIcon | True |

Событийно-управляемые процедуры в форме Авторизация представлены в приложении 7 таблице 9

Таблица 9 – Событийно-управляемые процедуры в форме «Авторизация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| AuthorizationForm | public AuthorizationForm()  {  InitializeComponent();  }  int k = 0; |
| RemoveButton\_Click | private void RemoveButton\_Click(object sender, EventArgs e)  {  if (LoginTextBox1.Text == "" && PassTextBox1.Text == "")  {  MessageBox.Show("Поля для ввода пусты!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  LoginTextBox1.Text = "";  PassTextBox1.Text = "";  } |
| Form2\_Load | private void Form2\_Load(object sender, EventArgs e)  {  PassTextBox1.PasswordChar = '•';  ToolTip t = new ToolTip();  t.SetToolTip(RemoveButton, "Очистка полей");  t.SetToolTip(CloseButton, "Закрыть форму");  t.SetToolTip(ShowPasswordAuth, "Показать пароль");  t.SetToolTip(HideButtonAuth, "Спрятать пароль");  t.SetToolTip(LoginTextBox1, "Введите логин");  t.SetToolTip(PassTextBox1, "Введите пароль");  } |
| EnterButton\_Click | private void EnterButton\_Click(object sender, EventArgs e)  {  k++;  if (k > 5)  {  MessageBox.Show("Исчерпано количество попыток авторизации", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);  Close();  }  label4.Text = "Попыток входа: " + k.ToString();  SqlConnection sqlConnect = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True");  SqlCommand work = new SqlCommand("Select Rights, Famnam, Name From Users where Login = @login", sqlConnect);  work.Parameters.AddWithValue("@login", LoginTextBox1.Text);  sqlConnect.Open();  SqlDataAdapter da = new SqlDataAdapter("select \* from Users", sqlConnect);  DataTable dt = new DataTable();  da.Fill(dt);  Boolean flag = false;  for (int i = 0; i < dt.Rows.Count; i++)  if ((dt.Rows[i]["Login"].ToString() == LoginTextBox1.Text) && (dt.Rows[i]["Password"].ToString() == PassTextBox1.Text))  flag = true;  if (flag == true)  {  MessageBox.Show("Вы успешно вошли!", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);  this.Hide();  var r = work.ExecuteScalar().ToString();  if (r == "Бухгалтер")  {  Form1 UsrFrm = new Form1();  UsrFrm.Show();  SqlDataReader rd = work.ExecuteReader();  UsrFrm.tabPage5.Parent = null;  UsrFrm.InfoButton.Visible = false;  UsrFrm.AdminButton.Visible = false;  UsrFrm.AddButtonDep.Visible = false;  UsrFrm.EditButtonDep.Visible = false;  UsrFrm.RemoveButtonDep.Visible = false;  UsrFrm.AddButtonTech.Visible = false;  UsrFrm.EditButtonTech.Visible = false;  UsrFrm.RemoveButtonTech.Visible = false;  UsrFrm.AddButtonExecut.Visible = false;  UsrFrm.EditButtonExecut.Visible = false;  UsrFrm.RemoveButtonExecut.Visible = false;  UsrFrm.AddButtonChief.Visible = false;  UsrFrm.EditButtonChief.Visible = false;  UsrFrm.RemoveButtonChief.Visible = false;  while (rd.Read())  {  UsrFrm.UserName.Text = String.Format("{0} {1}", rd["Famnam"], rd["Name"]);  UsrFrm.UserRight.Text = String.Format("{0}", rd["Rights"]);  }  }  if (r == "Начальник")  {  Form1 UsrFrm = new Form1();  UsrFrm.Show();  SqlDataReader rd = work.ExecuteReader();  UsrFrm.tabPage5.Parent = null;  UsrFrm.InfoButton.Visible = false;  UsrFrm.AdminButton.Visible = false;  UsrFrm.AddButtonCust.Visible = false;  UsrFrm.EditButtonCust.Visible = false;  UsrFrm.RemoveButtonCust.Visible = false;  UsrFrm.AddButtonApp.Visible = false;  UsrFrm.EditButtonApp.Visible = false;  UsrFrm.RemoveButtonApp.Visible = false;  UsrFrm.AddButtonChief.Visible = false;  UsrFrm.EditButtonChief.Visible = false;  UsrFrm.RemoveButtonChief.Visible = false;  UsrFrm.dataGridView5.Visible = false;  UsrFrm.RefreshButtonChief.Visible = false;  while (rd.Read())  {  UsrFrm.UserName.Text = String.Format("{0} {1}", rd["Famnam"], rd["Name"]);  UsrFrm.UserRight.Text = String.Format("{0}", rd["Rights"]);  }  }  if (r == "Администратор")  {  Form1 UsrFrm = new Form1();  UsrFrm.Show();  SqlDataReader rd = work.ExecuteReader();  UsrFrm.InfoButton.Visible = false;  while (rd.Read())  {  UsrFrm.UserName.Text = String.Format("{0} {1}", rd["Famnam"], rd["Name"]);  UsrFrm.UserRight.Text = String.Format("{0}", rd["Rights"]);  }  }  if (r == "Сотрудник")  {  Form1 UsrFrm = new Form1();  UsrFrm.Show();  SqlDataReader rd = work.ExecuteReader();  UsrFrm.tabPage1.Parent = null;  UsrFrm.tabPage3.Parent = null;  UsrFrm.tabPage4.Parent = null;  UsrFrm.tabPage5.Parent = null;  UsrFrm.AddButtonApp.Visible = false;  UsrFrm.EditButtonApp.Visible = false;  UsrFrm.RemoveButtonApp.Visible = false;  UsrFrm.InfoButton.Visible = false;  UsrFrm.AdminButton.Visible = false;  while (rd.Read())  {  UsrFrm.UserName.Text = String.Format("{0} {1}", rd["Famnam"], rd["Name"]);  UsrFrm.UserRight.Text = String.Format("{0}", rd["Rights"]);  }  }  if (r == "Гость")  {  Form1 UsrFrm = new Form1();  UsrFrm.Show();  SqlDataReader rd = work.ExecuteReader();  UsrFrm.tabPage1.Parent = null;  UsrFrm.tabPage2.Parent = null;  UsrFrm.tabPage3.Parent = null;  UsrFrm.tabPage4.Parent = null;  UsrFrm.AddButtonCompApp.Visible = false;  UsrFrm.EditButtonCompApp.Visible = false;  UsrFrm.RemoveButtonCompApp.Visible = false;  UsrFrm.AdminButton.Visible = false;  while (rd.Read())  {  UsrFrm.UserName.Text = String.Format("{0} {1}", rd["Famnam"], rd["Name"]);  UsrFrm.UserRight.Text = String.Format("{0}", rd["Rights"]);  }  }  }  else  if (LoginTextBox1.Text == "" && PassTextBox1.Text == "")  {  MessageBox.Show("Не введены данные в оба текстовые поля!", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);  }  else  if (LoginTextBox1.Text == "" || PassTextBox1.Text == "")  {  MessageBox.Show("Не введены данные в одно поле!", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);  }  else  if ((flag == false))  {  MessageBox.Show("Не совпадает пара логин/пароль с данными в базе данных!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  } |
| RegistrationLinkLabel\_LinkClicked | private void RegisterLinkLabel\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)  {  this.Hide();  RegistrationForm RegForm = new RegistrationForm();  RegForm.Show();  } |
| CloseButton\_Click | private void CloseButton\_Click(object sender, EventArgs e)  {  this.Close();  } |
| LoginTextBox1\_KeyPress | private void LoginTextBox1\_KeyPress(object sender, KeyPressEventArgs e)  {  if ((e.KeyChar < 'а' || e.KeyChar > 'я') && (e.KeyChar < 'А' || e.KeyChar > 'Я') && e.KeyChar != 'ё' && e.KeyChar != 'Ё')  {  }  else  e.Handled = true;  if (Char.IsSymbol(e.KeyChar) || Char.IsPunctuation(e.KeyChar) || Char.IsSeparator(e.KeyChar))  {  e.Handled = true;  }  else  if (e.KeyChar != 8)  {  if (LoginTextBox1.TextLength >= 20)  {  MessageBox.Show("Логин может содержать максимум 20 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| PassTextBox1\_KeyPress | private void PassTextBox1\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar != 8)  {  if (PassTextBox1.TextLength >= 8)  {  MessageBox.Show("Пароль может содержать максимум 8 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| HideButtonAuth | private void HideButtonAuth\_Click(object sender, EventArgs e)  {  HideButtonAuth.Visible = false;  ShowPasswordAuth.Visible = true;  PassTextBox1.PasswordChar = '•';  } |
| ShowPasswordAuth | private void ShowPasswordAuth\_Click(object sender, EventArgs e)  {  HideButtonAuth.Visible = true;  ShowPasswordAuth.Visible = false;  PassTextBox1.PasswordChar = (char)0;  } |

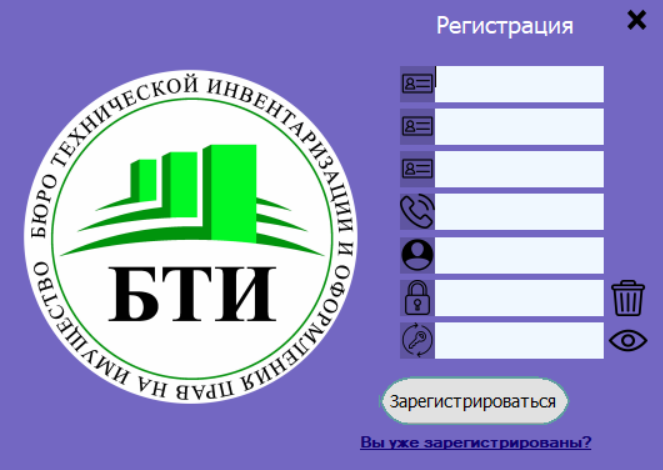


Рисунок 17 – Форма “Регистрация”

Свойства элементов формы Регистрация представлены в приложении 7 в таблице 10.

Таблица 10 – Свойства элементов формы Регистрация

| Объект | Свойство | Значение |
| --- | --- | --- |
| Регистрация | FormBorderStyle | None |
| Text | Регистрация |
| StartPosition | CentreScreen |
| Size | 710; 471 |
| ShowIcon | True |

Событийно-управляемые процедуры в форме Регистрация представлены в приложении 7 таблице 11

Таблица 11 – Событийно-управляемые процедуры в форме «Регистрация»

| Объект | Значение |
| --- | --- |
| Class Registration | public RegistrationForm()  {  InitializeComponent();  } |
| checkuser | private Boolean checkuser()  {  SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True");  var loginusr = LoginTextBox.Text;  SqlDataAdapter adapter = new SqlDataAdapter();  DataTable table = new DataTable();  string querystring = $"SELECT Login FROM Users WHERE Login = '{loginusr}'";  SqlCommand cmd = new SqlCommand(querystring, cn);  adapter.SelectCommand = cmd;  adapter.Fill(table);  if(table.Rows.Count > 0)  {  MessageBox.Show("Пользователь с таким логином уже существует!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);  return true;  }  else  {  return false;  }  } |
| Form3\_Load | private void Form3\_Load(object sender, EventArgs e)  {  PassTextBox.PasswordChar = '•';  ConfPassTextBox.PasswordChar = '•';  ToolTip t = new ToolTip();  t.SetToolTip(RemoveButton, "Очистка полей");  t.SetToolTip(CloseButton, "Закрытие формы");  t.SetToolTip(ShowPasswordReg, "Показать пароль");  t.SetToolTip(HideButtonReg, "Спрятать пароль");  t.SetToolTip(FamNameTextBox, "Введите фамилию");  t.SetToolTip(NameTextBox, "Введите имя");  t.SetToolTip(SurnameTextBox, "Введите отчество");  t.SetToolTip(PhoneTextBox, "Введите номер телефона");  t.SetToolTip(LoginTextBox, "Введите логин");  t.SetToolTip(PassTextBox, "Введите пароль");  t.SetToolTip(ConfPassTextBox, "Введите пароль повторно");  } |
| CloseButton\_Click | private void CloseButton\_Click(object sender, EventArgs e)  {  this.Close();  } |
| RegistrButton\_Click | private void RegistrButton\_Click(object sender, EventArgs e)  {  SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True");  var login = LoginTextBox.Text;  var password = PassTextBox.Text;  var famnam = FamNameTextBox.Text;  var name = NameTextBox.Text;  var surname = SurnameTextBox.Text;  var phone = PhoneTextBox.Text;  SqlCommand cmd;  string querystring = $"insert into Users(Login, Password, Famnam, Name, Surname, Phone) values('{login}', '{password}', '{famnam}', '{name}', '{surname}', '{phone}')";  cmd = new SqlCommand(querystring, cn);  cn.Open();  if (FamNameTextBox.Text == "" || NameTextBox.Text == "" || SurnameTextBox.Text == "" || PhoneTextBox.Text == "")  {  MessageBox.Show("У вас есть незаполненные поля!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  if (LoginTextBox.Text == "")  {  MessageBox.Show("Введите логин!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  if (PassTextBox.Text == "")  {  MessageBox.Show("Введите пароль!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  if (ConfPassTextBox.Text == "")  {  MessageBox.Show("Введите подтверждение пароля!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  if (PassTextBox.Text != ConfPassTextBox.Text)  {  MessageBox.Show("Неправильно введено подтверждение пароля!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  if (checkuser())  {  return;  }  else  if (cmd.ExecuteNonQuery() == 1)  {  MessageBox.Show("Аккаунт успешно создан!", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);  this.Hide();  AuthorizationForm LogForm = new AuthorizationForm();  LogForm.Show();  }  else  {  MessageBox.Show("Аккаунт не создан!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  } |
| FamTextBox\_KeyPress | private void FamNameTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if ((e.KeyChar < 'a' || e.KeyChar > 'z') && (e.KeyChar < 'A' || e.KeyChar > 'Z'))  {  }  else  e.Handled = true;  if (Char.IsSymbol(e.KeyChar) || Char.IsPunctuation(e.KeyChar) || Char.IsSeparator(e.KeyChar))  {  e.Handled = true;  }  else  if (Char.IsDigit(e.KeyChar) && e.KeyChar != 8)  {  e.Handled = true;  }  else  if (e.KeyChar != 8)  {  if (FamNameTextBox.TextLength >= 20)  {  MessageBox.Show("Данное поле может содержать только 20 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| NameTextBox\_KeyPress | private void NameTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if ((e.KeyChar < 'a' || e.KeyChar > 'z') && (e.KeyChar < 'A' || e.KeyChar > 'Z'))  {  }  else  e.Handled = true;  if (Char.IsSymbol(e.KeyChar) || Char.IsPunctuation(e.KeyChar) || Char.IsSeparator(e.KeyChar))  {  e.Handled = true;  }  else  if (Char.IsDigit(e.KeyChar) && e.KeyChar != 8)  {  e.Handled = true;  }  else  if (e.KeyChar != 8)  {  if (NameTextBox.TextLength >= 20)  {  MessageBox.Show("Данное поле может содержать только 20 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| SurnameTextBox\_KeyPress | private void SurnameTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if ((e.KeyChar < 'a' || e.KeyChar > 'z') && (e.KeyChar < 'A' || e.KeyChar > 'Z'))  {  }  else  e.Handled = true;  if (Char.IsSymbol(e.KeyChar) || Char.IsPunctuation(e.KeyChar) || Char.IsSeparator(e.KeyChar))  {  e.Handled = true;  }  else  if (Char.IsDigit(e.KeyChar) && e.KeyChar != 8)  {  e.Handled = true;  }  else  if (e.KeyChar != 8)  {  if (SurnameTextBox.TextLength >= 20)  {  MessageBox.Show("Данное поле может содержать только 20 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| PhoneTextBox\_KeyPress | private void PhoneTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if (!Char.IsDigit(e.KeyChar) && e.KeyChar != 8)  {  e.Handled = true;  }  else  if (e.KeyChar != 8)  {  if (PhoneTextBox.TextLength >= 7)  {  MessageBox.Show("Данное поле может содержать только 7 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| LoginTextBox\_KeyPress | private void LoginTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if ((e.KeyChar < 'а' || e.KeyChar > 'я') && (e.KeyChar < 'А' || e.KeyChar > 'Я') && e.KeyChar != 'ё' && e.KeyChar != 'Ё')  {  }  else  e.Handled = true;  if (Char.IsSymbol(e.KeyChar) || Char.IsPunctuation(e.KeyChar) || Char.IsSeparator(e.KeyChar))  {  e.Handled = true;  }  else  if (e.KeyChar != 8)  {  if (LoginTextBox.TextLength >= 20)  {  MessageBox.Show("Данное поле может содержать только 20 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| PassTextBox\_KeyPress | private void PassTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar != 8)  {  if (PassTextBox.TextLength >= 8)  {  MessageBox.Show("Данное поле может содержать только 8 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| ConfPassTextBox | private void ConfPassTextBox\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar != 8)  {  if (ConfPassTextBox.TextLength >= 8)  {  MessageBox.Show("Данное поле может содержать только 8 символов", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Information);  e.Handled = true;  }  }  } |
| RemoveButton\_Click | private void RemoveButton\_Click(object sender, EventArgs e)  {  if (FamNameTextBox.Text == "" && NameTextBox.Text == "" && PhoneTextBox.Text == "" && SurnameTextBox.Text == "" && LoginTextBox.Text == "" && PassTextBox.Text == "" && ConfPassTextBox.Text == "")  {  MessageBox.Show("Поля для ввода пусты!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  FamNameTextBox.Text = "";  NameTextBox.Text = "";  PhoneTextBox.Text = "";  SurnameTextBox.Text = "";  LoginTextBox.Text = "";  PassTextBox.Text = "";  ConfPassTextBox.Text = "";  } |
| AuthLinkLabel\_LinkClicked | private void AuthLinkLabel\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)  {  this.Hide();  AuthorizationForm LogForm = new AuthorizationForm();  LogForm.Show();  } |
| HideButtonReg\_Click | private void HideButtonReg\_Click(object sender, EventArgs e)  {  HideButtonReg.Visible = false;  ShowPasswordReg.Visible = true;  PassTextBox.PasswordChar = '•';  ConfPassTextBox.PasswordChar = '•';  } |
| ShowPasswordReg\_Click | private void ShowPasswordReg\_Click(object sender, EventArgs e)  {  ShowPasswordReg.Visible=false;  HideButtonReg.Visible = true;  PassTextBox.PasswordChar = (char)0;  ConfPassTextBox.PasswordChar = (char)0;  } |

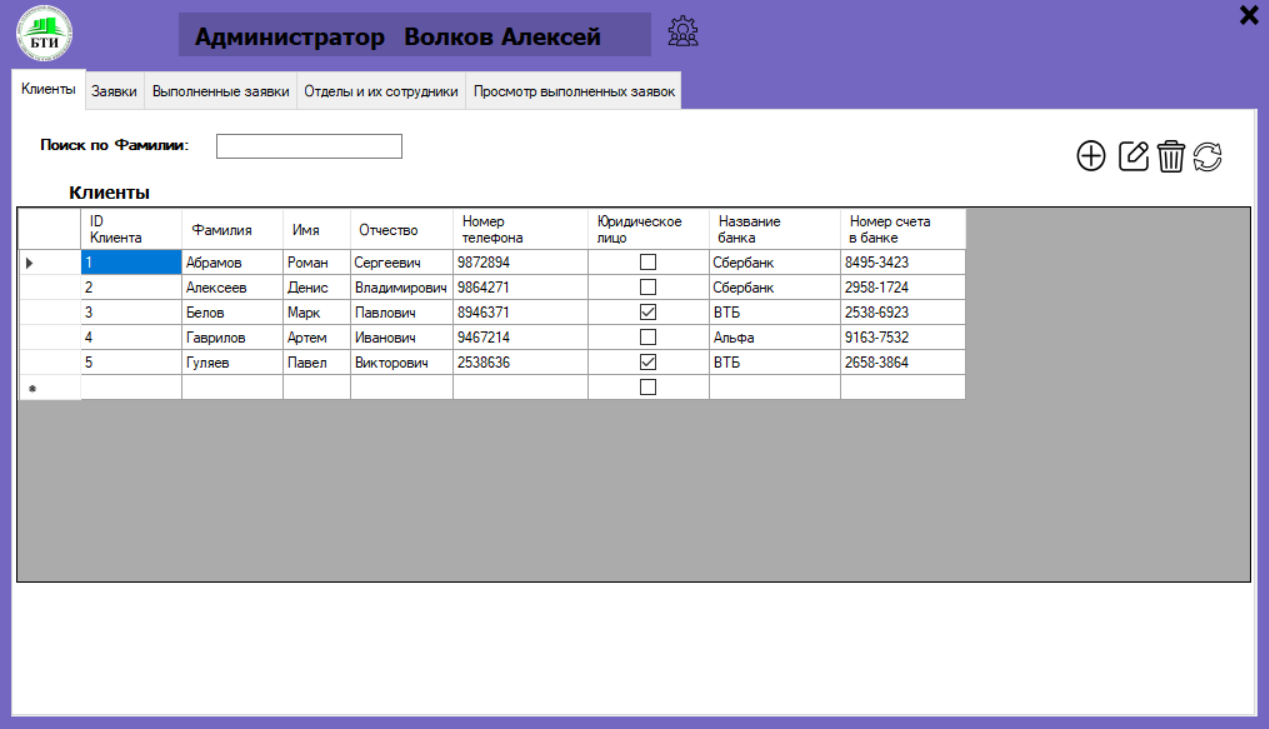


Рисунок 18 – Главная форма

Свойство элементов главной формы представлены в приложении 7 в таблице 12

Таблица 12 – Свойства элементов Главной формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Главная форма | FormBorderStyle | None |
| Text | Главная форма |
| StartPosition | CentreScreen |
| Size | 1354; 722 |
| ShowIcon | True |

Событийно-управляемые процедуры в Главной форме представлены в приложении 7 таблице 13.

Таблица 13 – Событийно-управляемые процедуры в форме «Главная форма»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Class Form1 | public Form1()  {  InitializeComponent();  } |
| Form1\_Load | private void Form1\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.TechnicalWorkers". При необходимости она может быть перемещена или удалена.  this.technicalWorkersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.TechnicalWorkers);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.Executors". При необходимости она может быть перемещена или удалена.  this.executorsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Executors);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.HeadsOfDepartments". При необходимости она может быть перемещена или удалена.  this.headsOfDepartmentsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.HeadsOfDepartments);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.TechnicalWorkers". При необходимости она может быть перемещена или удалена.  this.technicalWorkersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.TechnicalWorkers);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.Executors". При необходимости она может быть перемещена или удалена.  this.executorsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Executors);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.HeadsOfDepartments". При необходимости она может быть перемещена или удалена.  this.headsOfDepartmentsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.HeadsOfDepartments);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.CompletedApplications". При необходимости она может быть перемещена или удалена.  this.completedApplicationsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.CompletedApplications);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.CompletedApplications". При необходимости она может быть перемещена или удалена.  this.completedApplicationsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.CompletedApplications);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.Departments". При необходимости она может быть перемещена или удалена.  this.departmentsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Departments);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.Applications". При необходимости она может быть перемещена или удалена.  this.applicationsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Applications);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВDataSet.Customers". При необходимости она может быть перемещена или удалена.  this.customersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Customers);  ToolTip t = new ToolTip();  t.SetToolTip(CloseButton, "Закрытие формы");  t.SetToolTip(InfoButton, "Доп.информация");  t.SetToolTip(AdminButton, "Редактирование пользователей ИС");  t.SetToolTip(AddButtonCust, "Добавить клиента");  t.SetToolTip(EditButtonCust, "Редактировать клиента");  t.SetToolTip(RemoveButtonCust, "Удалить клиента");  t.SetToolTip(RefreshButtonCust, "Обновить таблицу");  t.SetToolTip(AddButtonApp, "Добавить заявку");  t.SetToolTip(EditButtonApp, "Редактировать заявку");  t.SetToolTip(RemoveButtonApp, "Удалить заявку");  t.SetToolTip(RefreshButtonApp, "Обновить таблицу");  t.SetToolTip(AddButtonCompApp, "Добавить выполненную заявку");  t.SetToolTip(EditButtonCompApp, "Редактировать выполненную заявку");  t.SetToolTip(RemoveButtonCompApp, "Удалить выполненную заявку");  t.SetToolTip(RefreshButtonCompApp, "Обновить таблицу");  t.SetToolTip(AddButtonDep, "Добавить отдел");  t.SetToolTip(EditButtonDep, "Редактировать отдел");  t.SetToolTip(RemoveButtonDep, "Удалить отдел");  t.SetToolTip(RefreshButtonDep, "Обновить таблицу");  t.SetToolTip(AddButtonTech, "Добавить техника");  t.SetToolTip(EditButtonTech, "Редактировать техника");  t.SetToolTip(RemoveButtonTech, "Удалить техника");  t.SetToolTip(RefreshButtonTech, "Обновить таблицу");  t.SetToolTip(AddButtonExecut, "Добавить исполнителя");  t.SetToolTip(EditButtonExecut, "Редактировать исполнителя");  t.SetToolTip(RemoveButtonExecut, "Удалить исполнителя");  t.SetToolTip(RefreshButtonExecut, "Обновить таблицу");  t.SetToolTip(AddButtonChief, "Добавить начальника");  t.SetToolTip(EditButtonChief, "Редактировать начальника");  t.SetToolTip(RemoveButtonChief, "Удалить начальника");  t.SetToolTip(RefreshButtonChief, "Обновить таблицу");  t.SetToolTip(RefreshButtonGuest, "Обновить таблицу");  } |
| CloseButton\_Click | private void CloseButton\_Click(object sender, EventArgs e)  {  this.Close();  AuthorizationForm AuthFrm = new AuthorizationForm();  AuthFrm.ShowDialog();  } |
| RefrButton\_Click | // Таблица "Отделы"  private void RefrButton\_Click(object sender, EventArgs e)  {  this.departmentsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Departments);  this.executorsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Executors);  this.headsOfDepartmentsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.HeadsOfDepartments);  this.technicalWorkersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.TechnicalWorkers);  } |
| AddButtonCust\_Click | // Кнопки таблицы "Клиенты"  private void AddButtonCust\_Click(object sender, EventArgs e)  {  AddCust RedCustForm = new AddCust();  RedCustForm.ShowDialog();  } |
| EditButtonCust\_Click | private void EditButtonCust\_Click(object sender, EventArgs e)  {  EditCust fr = new EditCust();  int row = dataGridView1.CurrentRow.Index;  fr.textBox9.Text = dataGridView1[0, row].Value.ToString();  fr.textBox1.Text = dataGridView1[1, row].Value.ToString();  fr.textBox2.Text = dataGridView1[2, row].Value.ToString();  fr.textBox3.Text = dataGridView1[3, row].Value.ToString();  fr.textBox4.Text = dataGridView1[4, row].Value.ToString();  if (dataGridView1[5, row].Value.Equals(true))  fr.checkBox1.Checked = true;  else  fr.checkBox1.Checked = false;  fr.textBox5.Text = dataGridView1[6, row].Value.ToString();  fr.textBox6.Text = dataGridView1[7, row].Value.ToString();  fr.ShowDialog();  } |
| RemoveButtonCust\_Click | private void RemoveButtonCust\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM Customers WHERE Customer\_Code=@id", cn);  int id = int.Parse(dataGridView1.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| AddButtonApp\_Click | // Кнопки таблицы "Заявки"  private void AddButtonApp\_Click(object sender, EventArgs e)  {  AddApp AddAppForm = new AddApp();  AddAppForm.ShowDialog();  } |
| EditButtonApp\_Click | private void EditButtonApp\_Click(object sender, EventArgs e)  {  EditApp app = new EditApp();  int row = dataGridView2.CurrentRow.Index;  app.textBox3.Text = dataGridView2[0, row].Value.ToString();  app.comboBox1.Text = dataGridView2[1, row].Value.ToString();  app.textBox1.Text = dataGridView2[2, row].Value.ToString();  app.comboBox2.Text = dataGridView2[3, row].Value.ToString();  app.textBox6.Text = dataGridView2[7, row].Value.ToString();  app.textBox8.Text = dataGridView2[5, row].Value.ToString();  app.textBox9.Text = dataGridView2[6, row].Value.ToString();  if (dataGridView2[4, row].Value.Equals(true))  app.checkBox1.Checked = true;  else  app.checkBox1.Checked = false;  app.ShowDialog();  } |
| RemoveButtonApp\_Click | private void RemoveButtonApp\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM Applications WHERE Application\_Code=@id", cn);  int id = int.Parse(dataGridView2.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| AddButtonCompApp\_Click | // Кнопки таблицы "Выполненные заказы"  private void AddButtonCompApp\_Click(object sender, EventArgs e)  {  AddCompleteApp AddFrm = new AddCompleteApp();  AddFrm.ShowDialog();  } |
| EditButtonComp\_Add | private void EditButtonCompApp\_Click(object sender, EventArgs e)  {  EditCompleteApp EditFrm = new EditCompleteApp();  int row = dataGridView3.CurrentRow.Index;  EditFrm.comboBox1.Text = dataGridView3[1, row].Value.ToString();  EditFrm.comboBox2.Text = dataGridView3[2, row].Value.ToString();  EditFrm.textBox8.Text = dataGridView3[0, row].Value.ToString();  EditFrm.textBox1.Text = dataGridView3[7, row].Value.ToString();  EditFrm.textBox2.Text = dataGridView3[8, row].Value.ToString();  EditFrm.textBox3.Text = dataGridView3[11, row].Value.ToString();  if (dataGridView3[6, row].Value.Equals(true))  EditFrm.checkBox1.Checked = true;  else  EditFrm.checkBox1.Checked = false;  if (dataGridView3[10, row].Value.Equals(true))  EditFrm.checkBox2.Checked = true;  else  EditFrm.checkBox2.Checked = false;  EditFrm.textBox7.Text = dataGridView3[9, row].Value.ToString();  EditFrm.textBox4.Text = dataGridView3[3, row].Value.ToString();  EditFrm.textBox5.Text = dataGridView3[4, row].Value.ToString();  EditFrm.textBox6.Text = dataGridView3[5, row].Value.ToString();  EditFrm.ShowDialog();  } |
| RemoveButtonCompApp\_Click | private void RemoveButtonCompApp\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM CompletedApplications WHERE Application\_Code=@id", cn);  int id = int.Parse(dataGridView3.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| InfoButton\_Click | private void InfoButton\_Click(object sender, EventArgs e) // Кнопка для доп.информирования пользователя  {  if (UserRight.Text.Contains("Гость"))  {  MessageBox.Show("У вас есть возможность просмотра выполненных заявок с помощью поиска. Если данные по заявке(ам) отсутствуют обратитесь к сотрудникам БТИ или по почте 'btiang@yandex.ru'", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  } |
| AdminButton\_Click | // Кнопка администратора для редактирования пользователей  private void AdminButton\_Click(object sender, EventArgs e)  {  AdminForm AdFrm = new AdminForm();  AdFrm.ShowDialog();  } |
| DataTable CustDataGridView | // Метод для поиска в таблице Клиенты  private DataTable CustDataGridView()  {  string query = "SELECT \* FROM Customers";  query += " WHERE Fam LIKE '%' + @Fam + '%'";  query += " OR @Fam = ''";  string con = @"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  using (SqlConnection cn = new SqlConnection(con))  {  using (SqlCommand cmd = new SqlCommand(query, cn))  {  cmd.Parameters.AddWithValue("@Fam", searchTextBoxCust.Text.Trim());  using (SqlDataAdapter sda = new SqlDataAdapter(cmd))  {  DataTable dt = new DataTable();  sda.Fill(dt);  return dt;  }  }  }  } |
| DataTable AppDataGridView | // Метод для поиска в таблице Заявки  private DataTable AppDataGridView()  {  string query = "SELECT \* FROM Applications";  query += " WHERE Customer\_Code LIKE '%' + @id + '%'";  query += " OR @id = ''";  string con = @"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  using (SqlConnection cn = new SqlConnection(con))  {  using (SqlCommand cmd = new SqlCommand(query, cn))  {  cmd.Parameters.AddWithValue("@id", searchTextBoxApp.Text.Trim());  using (SqlDataAdapter sda = new SqlDataAdapter(cmd))  {  DataTable dt = new DataTable();  sda.Fill(dt);  return dt;  }  }  }  } |
| DataTable CompAppDataGridView | private DataTable CompAppDataGridView()  {  string query = "SELECT \* FROM CompletedApplications";  query += " WHERE Application\_Code LIKE '%' + @id + '%'";  query += " OR @id = ''";  string con = @"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  using (SqlConnection cn = new SqlConnection(con))  {  using (SqlCommand cmd = new SqlCommand(query, cn))  {  cmd.Parameters.AddWithValue("@id", searchTextBoxCompApp.Text.Trim());  using (SqlDataAdapter sda = new SqlDataAdapter(cmd))  {  DataTable dt = new DataTable();  sda.Fill(dt);  return dt;  }  }  }  } |
| DataTable GuestDataGridView | // Метод для поиска в таблице Выполненные заявки(Клиенты)  private DataTable GuestDataGridView()  {  string query = "SELECT \* FROM CompletedApplications";  query += " WHERE Application\_Code LIKE '%' + @id + '%'";  query += " OR @id = ''";  string con = @"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  using (SqlConnection cn = new SqlConnection(con))  {  using (SqlCommand cmd = new SqlCommand(query, cn))  {  cmd.Parameters.AddWithValue("@id", searchTextBoxGuest.Text.Trim());  using (SqlDataAdapter sda = new SqlDataAdapter(cmd))  {  DataTable dt = new DataTable();  sda.Fill(dt);  return dt;  }  }  }  } |
| searchTextBoxCust\_KeyUp | private void searchTextBoxCust\_KeyUp(object sender, KeyEventArgs e)  {  dataGridView1.DataSource = this.CustDataGridView();  } |
| RefreshButtonCust\_Click | // Кнопка обновления таблицы Клиенты  private void RefreshButtonCust\_Click(object sender, EventArgs e)  {  SqlConnection cn = new SqlConnection();  cn.ConnectionString = $"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  cn.Open();  SqlCommand cmd = new SqlCommand("SELECT \* FROM Customers", cn);  SqlDataAdapter ad = new SqlDataAdapter(cmd);  DataTable dt = new DataTable();  ad.Fill(dt);  BindingSource bs = new BindingSource();  bs.DataSource = dt;  dataGridView1.DataSource = bs;  } |
| RefreshButtonApp\_Click | // Кнопка обновления таблицы Заявки  private void RefreshButtonApp\_Click(object sender, EventArgs e)  {  SqlConnection cn = new SqlConnection();  cn.ConnectionString = $"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  cn.Open();  SqlCommand cmd = new SqlCommand("SELECT \* FROM Applications", cn);  SqlDataAdapter ad = new SqlDataAdapter(cmd);  DataTable dt = new DataTable();  ad.Fill(dt);  BindingSource bs = new BindingSource();  bs.DataSource = dt;  dataGridView2.DataSource = bs;  } |
| RefreshButtonCompApp\_Click | // Кнопка обновления таблицы Выполненные заявки  private void RefreshButtonCompApp\_Click(object sender, EventArgs e)  {  SqlConnection cn = new SqlConnection();  cn.ConnectionString = $"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  cn.Open();  SqlCommand cmd = new SqlCommand("SELECT \* FROM CompletedApplications", cn);  SqlDataAdapter ad = new SqlDataAdapter(cmd);  DataTable dt = new DataTable();  ad.Fill(dt);  BindingSource bs = new BindingSource();  bs.DataSource = dt;  dataGridView3.DataSource = bs;  } |
| RefreshButtonDep\_Click | // Кнопка обновления таблицы Отделы  private void RefreshButtonDep\_Click(object sender, EventArgs e)  {  SqlConnection cn = new SqlConnection();  cn.ConnectionString = $"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  cn.Open();  SqlCommand cmd = new SqlCommand("SELECT \* FROM Departments", cn);  SqlDataAdapter ad = new SqlDataAdapter(cmd);  DataTable dt = new DataTable();  ad.Fill(dt);  BindingSource bs = new BindingSource();  bs.DataSource = dt;  dataGridView4.DataSource = bs;  } |
| RefreshButtonGuest\_Click | // Кнопка обновления таблицы Выполненные заявки(Клиентская)  private void RefreshButtonGuest\_Click(object sender, EventArgs e)  {  SqlConnection cn = new SqlConnection();  cn.ConnectionString = $"Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True";  cn.Open();  SqlCommand cmd = new SqlCommand("SELECT \* FROM CompletedApplications", cn);  SqlDataAdapter ad = new SqlDataAdapter(cmd);  DataTable dt = new DataTable();  ad.Fill(dt);  BindingSource bs = new BindingSource();  bs.DataSource = dt;  dataGridView8.DataSource = bs;  } |
| searchTextBoxApp\_KeyUp | // TextBox для поиска в таблице Заявки  private void searchTextBoxApp\_KeyUp(object sender, KeyEventArgs e)  {  dataGridView2.DataSource = this.AppDataGridView();  } |
| searchTextBoxCompApp\_KeyUp | // TextBox для поиска в таблице Выполненные заявки  private void searchTextBoxCompApp\_KeyUp(object sender, KeyEventArgs e)  {  dataGridView3.DataSource = this.CompAppDataGridView();  } |
| searchTextBoxGuest\_KeyUp | // TextBox для поиска в таблице Выполненные заявки(Клиент)  private void searchTextBoxGuest\_KeyUp(object sender, KeyEventArgs e)  {  dataGridView8.DataSource = this.GuestDataGridView();  } |
| AddButtonDep\_Click | // Кнопки для таблицы Отделы  private void AddButtonDep\_Click(object sender, EventArgs e)  {  AddDepartmentForm addDepForm = new AddDepartmentForm();  addDepForm.ShowDialog();  } |
| EditButtonDep\_Click | private void EditButtonDep\_Click(object sender, EventArgs e)  {  EditDepartmentForm EditFrm = new EditDepartmentForm();  int row = dataGridView4.CurrentRow.Index;  EditFrm.textBox1.Text = dataGridView4[0, row].Value.ToString();  EditFrm.comboBox1.Text = dataGridView4[1, row].Value.ToString();  EditFrm.comboBox2.Text = dataGridView4[2, row].Value.ToString();  EditFrm.comboBox3.Text = dataGridView4[3, row].Value.ToString();  EditFrm.ShowDialog();  } |
| RemoveButtonDep\_Click | private void RemoveButtonDep\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM Departments WHERE Department\_Code=@id", cn);  int id = int.Parse(dataGridView4.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| AddButtonTech\_Click | // Кнопки для таблицы Техники  private void AddButtonTech\_Click(object sender, EventArgs e)  {  AddTechForm addTechFrm = new AddTechForm();  addTechFrm.ShowDialog();  } |
| EditButtonTech\_Click | private void EditButtonTech\_Click(object sender, EventArgs e)  {  EditTechForm EditFrm = new EditTechForm();  int row = dataGridView7.CurrentRow.Index;  EditFrm.textBox1.Text = dataGridView7[1, row].Value.ToString();  EditFrm.textBox2.Text = dataGridView7[2, row].Value.ToString();  EditFrm.textBox3.Text = dataGridView7[3, row].Value.ToString();  EditFrm.textBox4.Text = dataGridView7[0, row].Value.ToString();  EditFrm.ShowDialog();  } |
| RemoveButtonTech\_Click | private void RemoveButtonTech\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM TechnicalWorkers WHERE Worker\_Code=@id", cn);  int id = int.Parse(dataGridView7.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| RefreshButtonTech\_Click | private void RefreshButtonTech\_Click(object sender, EventArgs e)  {  this.technicalWorkersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.TechnicalWorkers);  } |
| AddButtonExecut\_Click | // Кнопки таблицы Исполнители  private void AddButtonExecut\_Click(object sender, EventArgs e)  {  AddExecutorForm AddExecFrm = new AddExecutorForm();  AddExecFrm.ShowDialog();  } |
| EditButtonExecut\_Click | private void EditButtonExecut\_Click(object sender, EventArgs e)  {  EditExecutorForm EditFrm = new EditExecutorForm();  int row = dataGridView6.CurrentRow.Index;  EditFrm.textBox1.Text = dataGridView6[1, row].Value.ToString();  EditFrm.textBox2.Text = dataGridView6[2, row].Value.ToString();  EditFrm.textBox3.Text = dataGridView6[3, row].Value.ToString();  EditFrm.textBox4.Text = dataGridView6[0, row].Value.ToString();  EditFrm.ShowDialog();  } |
| RemoveButtonExecut\_Click | private void RemoveButtonExecut\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM Executors WHERE Executor\_Code=@id", cn);  int id = int.Parse(dataGridView6.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| RefreshButtonExecut\_Click | private void RefreshButtonExecut\_Click(object sender, EventArgs e)  {  this.executorsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.Executors);  } |
| AddButtonChief\_Click | // Кнопки таблицы Начальники  private void AddButtonChief\_Click(object sender, EventArgs e)  {  AddChiefForm AddChiefFrm = new AddChiefForm();  AddChiefFrm.ShowDialog();  } |
| EditButtonChief\_Click | private void EditButtonChief\_Click(object sender, EventArgs e)  {  EditChiefForm EditChiefFrm = new EditChiefForm();  int row = dataGridView5.CurrentRow.Index;  EditChiefFrm.textBox1.Text = dataGridView5[1, row].Value.ToString();  EditChiefFrm.textBox2.Text = dataGridView5[2, row].Value.ToString();  EditChiefFrm.textBox3.Text = dataGridView5[3, row].Value.ToString();  EditChiefFrm.textBox4.Text = dataGridView5[0, row].Value.ToString();  EditChiefFrm.ShowDialog();  } |
| RemoveButtonChief\_Click | private void RemoveButtonChief\_Click(object sender, EventArgs e)  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM HeadsOfDepartments WHERE Chief\_Code=@id", cn);  int id = int.Parse(dataGridView5.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |
| RefreshButtonChief\_Click | private void RefreshButtonChief\_Click(object sender, EventArgs e)  {  this.headsOfDepartmentsTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВDataSet.HeadsOfDepartments);  } |

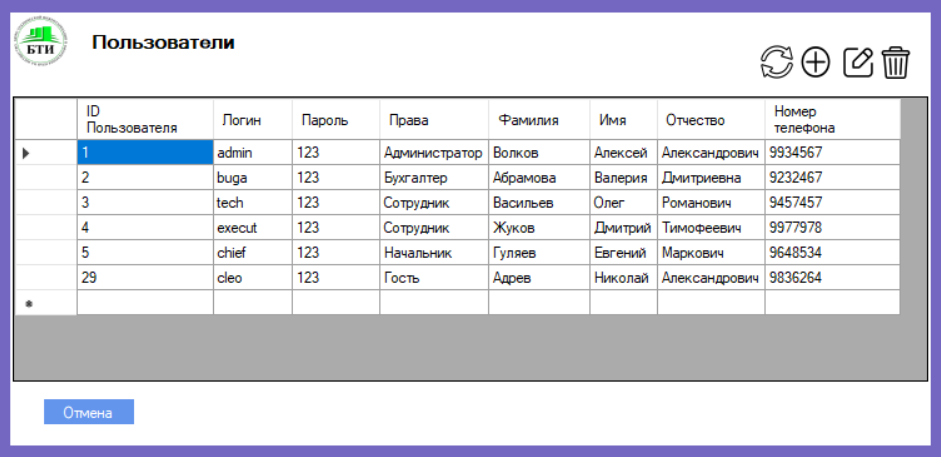


Рисунок 19 – Форма “Пользователи”

Свойства элементов формы Пользователи представлены в приложении 7 в таблице 14.

Таблица 14 – Свойства элементов формы Пользователи

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Пользователи | FormBorderStyle | None |
| Text | Пользователи |
| StartPosition | CentreScreen |
| Size | 1354; 722 |
| ShowIcon | True |

Событийно-управляемые процедуры в форме Пользователи представлены в приложении 7 таблице 15.

Таблица 15 – Событийно-управляемые процедуры в форме «Пользователи»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Class AdminForm | public AdminForm()  {  InitializeComponent();  } |
| AdminForm\_Load | private void AdminForm\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Иванченко\_ИВAdminDataSet.Users". При необходимости она может быть перемещена или удалена.  this.usersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВAdminDataSet.Users);  ToolTip t = new ToolTip();  t.SetToolTip(AddButtonAdmin, "Добавить пользователя");  t.SetToolTip(EditButtonAdmin, "Редактировать пользователя");  t.SetToolTip(RemoveButtonAdmin, "Удалить пользователя");  t.SetToolTip(RefreshButtonAdmin, "Обновить пользователя");  } |
| AdminCancelButton\_Click | // Кнопки формы администратора  private void AdminCancelButton\_Click(object sender, EventArgs e) // Кнопка возврата к главной форме  {  this.Close();  } |
| RefreshButtonAdmin\_Click | private void RefreshButtonAdmin\_Click(object sender, EventArgs e) // Кнопка обновления данных пользователей  {  this.usersTableAdapter.Fill(this.уП01\_ИСПП5\_Иванченко\_ИВAdminDataSet.Users);  } |
| AddButtonAdmin\_Click | private void AddButtonAdmin\_Click(object sender, EventArgs e) // Кнопка добавления пользователя  {  AddUser AddUsrFrm = new AddUser();  AddUsrFrm.ShowDialog();  } |
| EditButtonAdmin\_Click | private void EditButtonAdmin\_Click(object sender, EventArgs e) // Кнопка редактирования пользователя  {  EditUser EditUsrFrm = new EditUser();  int row = UsersDataGrid.CurrentRow.Index;  EditUsrFrm.comboBox1.Text = UsersDataGrid[3, row].Value.ToString();  EditUsrFrm.textBox7.Text = UsersDataGrid[0, row].Value.ToString();  EditUsrFrm.textBox1.Text = UsersDataGrid[1, row].Value.ToString();  EditUsrFrm.textBox2.Text = UsersDataGrid[2, row].Value.ToString();  EditUsrFrm.textBox3.Text = UsersDataGrid[4, row].Value.ToString();  EditUsrFrm.textBox4.Text = UsersDataGrid[5, row].Value.ToString();  EditUsrFrm.textBox5.Text = UsersDataGrid[6, row].Value.ToString();  EditUsrFrm.textBox6.Text = UsersDataGrid[7, row].Value.ToString();  EditUsrFrm.ShowDialog();  } |
| RemoveButtonAdmin\_Click | private void RemoveButtonAdmin\_Click(object sender, EventArgs e) // Кнопка удаления пользователя  {  using (SqlConnection cn = new SqlConnection("Data Source=DESKTOP-7J49MA8; Initial Catalog = УП01\_ИСПП5\_Иванченко\_ИВ; Integrated Security = True; MultipleActiveResultSets = True"))  {  SqlCommand com = new SqlCommand("DELETE FROM Users WHERE UserID=@id", cn);  int id = int.Parse(UsersDataGrid.CurrentRow.Cells[0].Value.ToString());  com.Parameters.AddWithValue("@id", id);  cn.Open();  try  {  com.ExecuteNonQuery();  MessageBox.Show("Запись удалена", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  catch  {  MessageBox.Show("Удалить не удалось!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  }  } |

Вывод по главе: была полностью спроектирована и разработана система базы данных, были полностью спроектированы и разработаны все программные модули информационной системы “Бюро технической инвентаризации”.

# 5 РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

Тестовый сценарий – это неавтоматизированный или автоматизированный сценарий, содержащий инструкции по реализации тестового набора.

## 5.1 Назначение эксперимента. Выбор и обоснование методики проведения тестирования

Назначение эксперимента: оценить производительность и надежность приложения при работе с большим объемом данных.

Выбор методики проведения тестирования: для оценки производительности и надежности приложения при работе с большим объемом данных рекомендуется использовать методику нагрузочного тестирования. Этот метод позволяет оценить, как приложение будет работать при максимальной нагрузке и определить его предельные возможности.

Обоснование выбора методики: Нагрузочное тестирование позволит оценить производительность приложения в условиях реальной нагрузки и выявить возможные узкие места, которые могут привести к сбоям или замедлению работы приложения. Также этот метод позволит оценить надежность приложения и его способность обрабатывать большие объемы данных без потери качества работы.

## 5.2 Технология тестирования

Для проведения нагрузочного тестирования можно использовать различные технологии, такие как Apache JMeter, LoadRunner, Gatling, и другие. Эти инструменты позволяют создавать сценарии нагрузочного тестирования, имитируя работу реальных пользователей или генерируя большое количество запросов к приложению.

Также для оценки производительности и надежности приложения можно использовать инструменты мониторинга, которые позволяют отслеживать работу приложения в реальном времени и выявлять узкие места и проблемы в его работе.

Выбор технологии зависит от конкретных требований и особенностей приложения, а также от доступных ресурсов и опыта команды тестировщиков.

## 5.3 Результаты проведения тестирования

В ходе тестирования был применен метод нагрузочного тестирования, который проверяет работу при нагрузке системы. Суть тестирования заключается в проверке работы всех обязательных функций.

Для проведения нагрузочного тестирования была составлена таблица 23, в которой указано количество запущенных копий программного продукта, нагрузка на процесс и оперативную память.

Таблица 23 – Результаты нагрузочного тестирования

|  |  |  |
| --- | --- | --- |
| Число копий | Нагрузка на процессор (%) | Нагрузка на память (%) |
| 1 | 0,2 | 0,4 |
| 2 | 0,5 | 1 |
| 5 | 3,6 | 3 |
| 8 | 4,8 | 7 |
| 10 | 6,8 | 12 |

Исходя из результатов тестирования, можно прийти к выводу, что нагрузка, оказываемая на центральный процессор и память, не имеет критичных значений. Это значит, что показатель производительности информационной системы на приемлемом уровне, то есть сбои и длительная загрузка не должны помешать пользователю при работе с данным программным продуктом.

Вывод по главе: было проведено тестирование информационной системы “Бюро технической инвентаризации”.

# ЗАКЛЮЧЕНИЕ

В результате учебной практики был проведен анализ предметной области, спроектирована и разработана информационная система «Бюро технической инвентаризации», серверная часть которой была реализована в MS SQL Server Management Studio, а клиентская часть в MS Visual Studio.

В MS SQL Server Management Studio была создана база данных для хранения информации о клиентах, заявках, отделах, выполненных заявках, начальников, исполнителей и техников.

Клиентская часть, реализованная в MS Visual Studio, отражает все хранящиеся в базе данных таблицы. Также разработаны некоторые запросы, позволяющие редактировать и удалять существующие данные, добавлять новые и осуществлять переходы по формам.

Цели учебной практики были достигнуты путем проведения анализа предметной области, проектирования информационной системы и разработки программных модулей. Это позволяет сделать вывод, что поставленные цели и задачи выполнены в полном объеме.

В дальнейшем данная информационная система может быть модернизирована путем добавления формы личного кабинета, создания версии для слабовидящих или редизайна и т.д.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. APET [Электронный ресурс]: URL: [https://a-pet.ru/moodle/course/view.php?id=2458](https://a-pet.ru/moodle/course/view.php?id=2458%20) (дата обращения:05.12.2023).
2. Cyberforum [Электронный ресурс]: URL: <cyberforum.ru> (дата обращения: 02.12.2023)
3. Youtube [Электронный ресурс]: URL: <https://www.youtube.com/watch?v=dQw4w9WgXcQ> (дата обращения: 06.12.2023)
4. Stackoverflow [Электронный ресурс]: URL: <stackoverflow.com> (дата обращения: 01.12.2023)
5. ChatGPT [Электронный ресурс]: URL: <https://chat.openai.com/> (дата обращения: 03.12.2023)
6. OtvetMail [Электронный ресурс]: URL: <otvet.mail.ru> (дата обращения 28.11.2023)
7. Mail [Электронный ресурс]: URL: <mail.ru> (дата обращения: 02.12.2023)
8. VK [Электронный ресурс]: URL: <https://vk.ru> (дата обращения: 06.12.2023)
9. GitHub [Электронный ресурс]: URL: <github.com> (дата обращения: 06.12.2023)
10. Habr [Электронный ресурс]: URL: <qna.habr.com> (дата обращения: 29.11.2023)