СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc152769764)

[1 ЦЕЛЬ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ 4](#_Toc152769765)

[1.1 Основные требования по технике безопасности во время прохождения учебной практики 4](#_Toc152769766)

[1.2 Нормативно-правовые документы 5](#_Toc152769767)

[2 ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ 7](#_Toc152769768)

[2.1 Описание предметной области 7](#_Toc152769769)

[2.2 Группы пользователей информационной системы 8](#_Toc152769770)

[2.3 Основные требования, предъявляемые к информационной системе 8](#_Toc152769771)

[3 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «БЮРО ТЕХНИЧЕСКОЙ ИНВЕНТАРИЗАЦИИ» 10](#_Toc152769772)

[3.1 Моделирование некоторых динамических аспектов системы 10](#_Toc152769773)

[3.1.1 Создание диаграммы вариантов использования 11](#_Toc152769774)

[3.1.2 Создание диаграммы деятельности 12](#_Toc152769775)

[3.2 Разработка функциональной структуры информационной системы. Карта навигации по информационной системе 15](#_Toc152769776)

[4 РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ (БЮРО ТЕХНИЧЕСКОЙ ИНВЕНТАРИЗАЦИИ) 18](#_Toc152769777)

[4.1 Моделирование и разработка базы данных 18](#_Toc152769778)

[4.2 Разработка программных модулей информационной системы «Бюро технической инвентаризации » 22](#_Toc152769779)

[5 РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ 51](#_Toc152769780)

[5.1 Назначение эксперимента. Выбор и обоснование методики проведения тестирования 51](#_Toc152769781)

[5.2 Технология тестирования 51](#_Toc152769782)

[5.3 Результаты проведения тестирования 52](#_Toc152769783)

[ЗАКЛЮЧЕНИЕ 54](#_Toc152769784)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 55](#_Toc152769785)

[ПРИЛОЖЕНИЕ A 56](#_Toc152769786)

ВВЕДЕНИЕ

Техническая инвентаризация нежилых помещений играет существенную роль в организации и обслуживании городской инфраструктуры. В связи с проведением сплошной инвентаризации в 2000 году в нашем городе, когда было выявлено более 16000 нежилых помещений разнообразного назначения (магазины, офисы, учреждения, мастерские и др.), становится ясной необходимость в разработке специализированного программного обеспечения для учета и обслуживания этой инфраструктуры.  
Каждое нежилое помещение имеет свою уникальную структуру, включая различные составные части (кабинеты, проходные, коридоры, и другие), что требует детального и точного учета. Для удобства и эффективности такой системы необходимо учитывать возможность привязки нескольких помещений к одному зданию, а также возможность существования нескольких составных частей в каждом помещении. Каждое здание обладает уникальным кадастровым номером, позволяющим однозначно определить его положение в городе. Разработка и внедрение программного обеспечения для учета и управления нежилыми помещениями позволит систематизировать данные об инвентаризации, упростить процессы обслуживания и обеспечить более точное планирование использования данного вида недвижимости. Такая система будет способствовать оптимизации работы отдела учета нежилых помещений Бюро технической инвентаризации, повышению эффективности процессов учета и обслуживания нежилой недвижимости в городе.

# 1 ЦЕЛЬ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ

Разработать прикладное программное обеспечение деятельности отдела учета нежилых помещений Бюро технической инвентаризации. Это магазины (встроенные, пристроенные и отдельно стоящие), офисы, учреждения, мастерские и т.д. Помещение может состоять из отдельных частей (кабинет, проходная, коридор). В одном здании может быть несколько помещений, а помещение может состоять из нескольких частей. Любое здание имеет уникальный кадастровый номер, однозначно определяющий его положение в городе.

## 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: Ознакомились с основной документацией

# 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

Отдел нежилых помещений Бюро технической инвентаризации. В 2000 г. в нашем городе была проведена сплошная инвентаризация, в ходе которой было выявлено 16000 нежилых помещений. Это магазины (встроенные, пристроенные и отдельно стоящие), офисы, учреждения, мастерские и т.д. Помещение может состоять из отдельных частей (кабинет, проходная, коридор). Составных частей помещений выявлено 265 000 шт. В одном здании может быть несколько помещений, а помещение может состоять из нескольких частей. Любое здание имеет уникальный кадастровый номер, однозначно определяющий его положение в городе.

Для учета и обработки данных об этих нежилых помещение отделу учета необходимо разработать специальное прикладное программное помещение. Это ПО должно обладать следующими возможностями:

Организация базы данных для хранения информации о нежилых помещениях, их составных частях и кадастровых номерах зданий.

* Добавление информации о каждом помещении, включая его тип, размеры, адрес и другие характеристики.
* Поиск и фильтрация помещение по различным параметрам, таким как тип помещения, его расположения и другие.
* Обновление информации о существующих помещениях при изменении их состояния или характеристики.

# 2.2 Группы пользователей информационной системы

Информационной системой будут пользоваться:

* администратор - имеет полный доступ к информационной системе;
* директор – может просматривать информацию;
* сотрудник - может вносить информацию о зданиях.

# 2.3 Основные требования, предъявляемые к информационной системе

Гибкость, способность к адаптации и дальнейшему развитию подразумевают возможность приспособления информационной системы к новым условиям, новым потребностям предприятия.

Выполнение этих условий возможно, если на этапе разработки информационной системы использовались общепринятые средства и методы документирования, так что по прошествии определенного времени сохранится возможность разобраться в структуре системы и внести в нее соответствующие изменения, даже если все разработчики или их часть по каким-либо причинам не смогут продолжить работу.

Надежность информационной системы подразумевает ее функционирование без искажения информации, потери данных по «техническим причинам».

Требование надежности обеспечивается созданием резервных копий хранимой информации, выполнения операций протоколирования, поддержанием качества каналов связи и физических носителей информации, использованием современных программных и аппаратных средств. Сюда же следует отнести защиту от случайных потерь информации в силу недостаточной квалификации персонала.

Система является эффективной, если с учетом выделенных ей ресурсов она позволяет решать возложенные на нее задачи в минимальные сроки.

В любом случае оценка эффективности будет производиться заказчиком, исходя из вложенных в разработку средств и соответствия представленной информационной системы его ожиданиям.

Под безопасностью, прежде всего, подразумевается свойство системы, в силу которого посторонние лица не имеют доступа к информационным ресурсам организации, кроме тех, которые для них предназначены.

Защита информации от постороннего доступа обеспечивается управлением доступом к ресурсам системы, использованием современных программных средств защиты информации. В крупных организациях целесообразно создавать подразделения, основным направлением деятельности которых было бы обеспечение информационной безопасности, в менее крупных организациях назначать сотрудника, ответственного за данный участок работы.

Вывод по главе: было описано, что такое предметная область и её описание; группы пользователей, которые будут пользоваться информационной системой «Бюро технической инвентаризации» 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.

В приложении А на рисунке 1 представлена диаграмма вариантов использования по информационной системе «Бюро технической инвентаризации »

# 3.1.1 Создание диаграммы вариантов использования

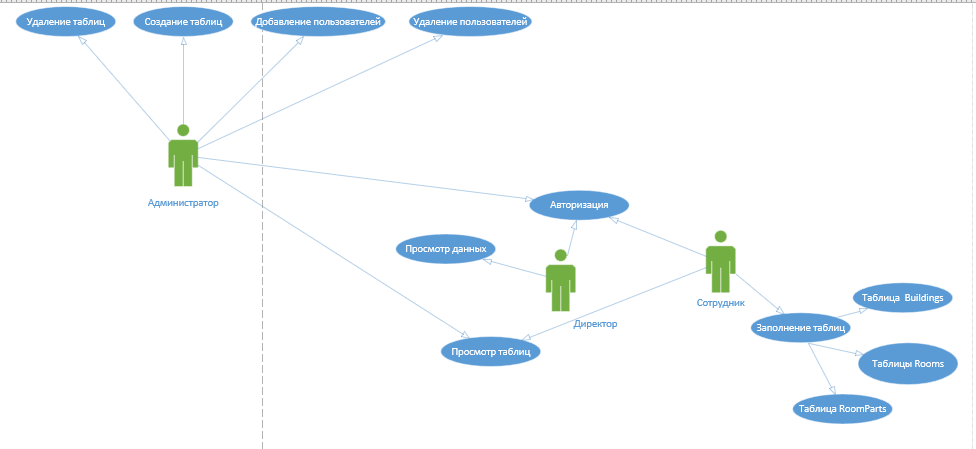


Рисунок 1 – Диаграмма вариантов использования

На диаграмме представлены такие актеры как администратор, директор, менеджер.

Администратор вправе не только просматривать, добавлять и редактировать данные, но и модифицировать информационную систему в целом. К таким модификациям можно отнести добавление новых таблиц в базу данных, удаление каких-либо полей из таблиц и прочее. Кроме того, администратор имеет права на разграничение прав доступа к базе данных. Это необходимо для того, чтобы сотрудники, не имеющие определенных полномочий, не могли получить доступ к различным объектам и данным.

Менеджер может просматривать данные всех таблиц, кроме таблицы «users», помимо просмотра есть возможность редактирования данных таблиц: «RoomParts», «Rooms», «Buildings».

Директор может просматривать все таблицы в базе данных.

# 3.1.2 Создание диаграммы деятельности

На рисунках 2 – 4 представлена диаграмма деятельности (по ролям) информационной системы «Бюро технической инвентаризации ».

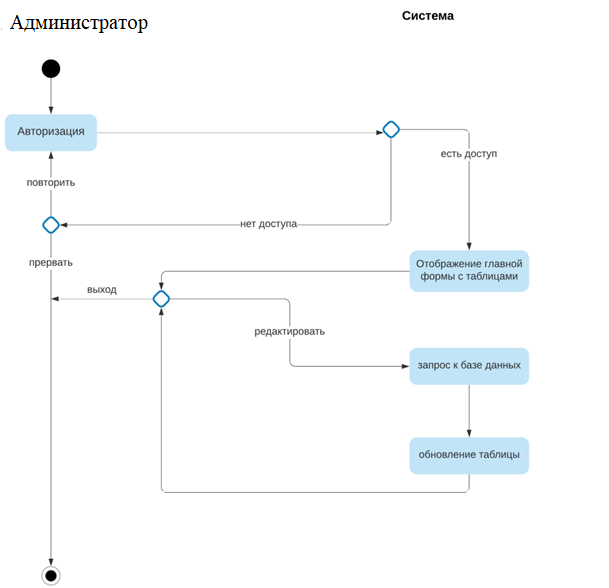


Рисунок 2 – Диаграмма деятельности (Администратор)

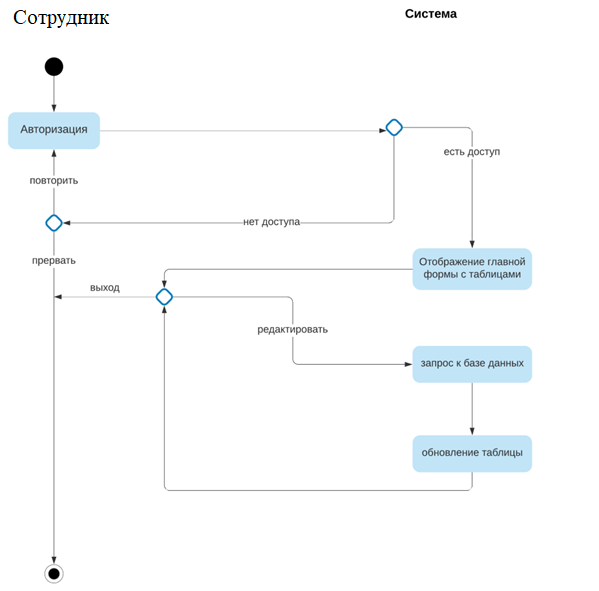


Рисунок 3 – Диаграмма деятельности (Сотрудник)

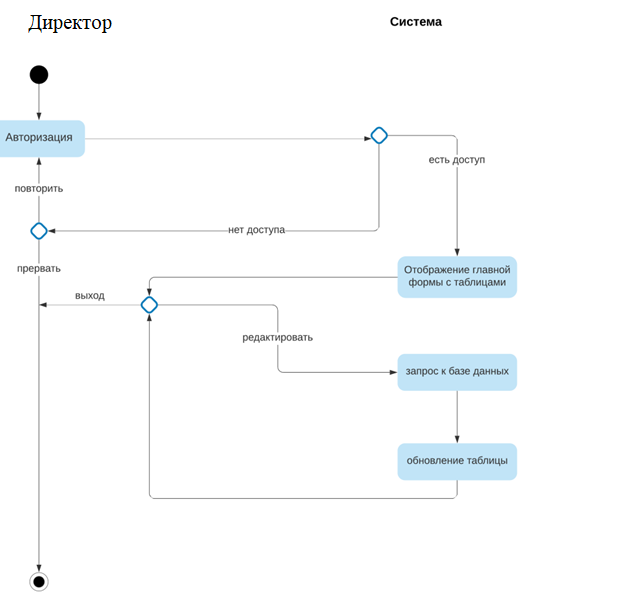


Рисунок 4 – Диаграмма деятельности (Директор)

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

В приложении А на рисунке 2 представлена карта переходов по информационной системе «Бюро технической инвентаризации »

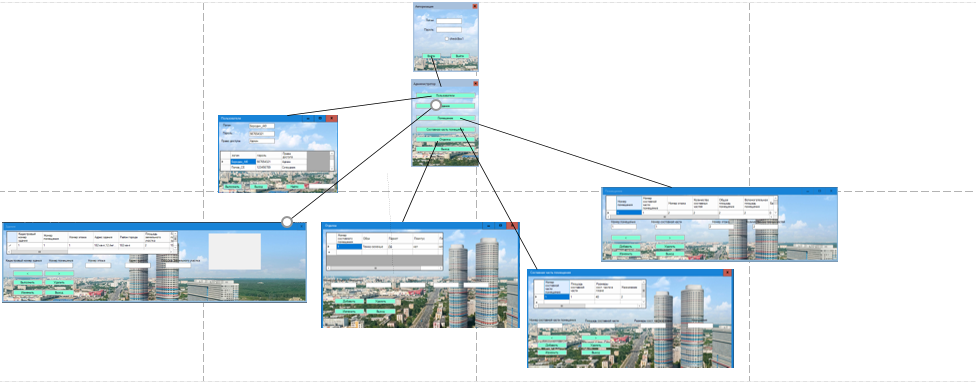


Рисунок 5 – Карта переходов

3.3 Разработка макета информационной системы

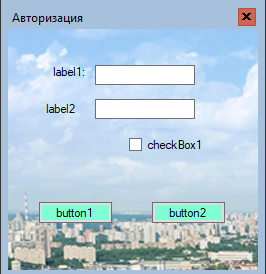


Рисунок 6 – Окно авторизации

На этапе авторизации пользователь может ввести логин и пароль, чтобы продолжить работу в информационной системе.

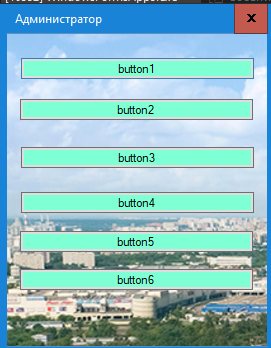


Рисунок 7 – Окно Администратора

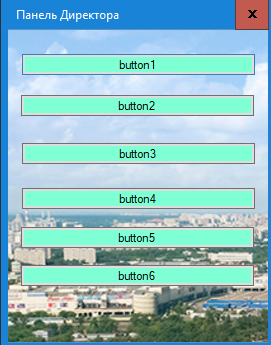


Рисунок 8 – Окно Директора

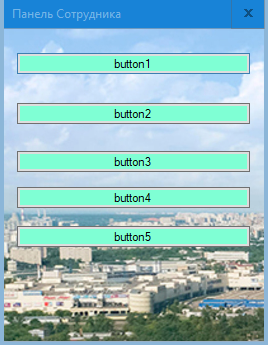


Рисунок 9 – Окно Сотрудника

Аналогичным образом в зависимости от роли пользователя выглядит главная форма, на которой расположены все необходимые таблицы. Например, администратор сможет изменять все таблицы, а для сотрудника будет доступно только изменять некоторые таблицы.

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

# 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.

## Моделирование и разработка базы данных

Исходя из анализа предметной области, можно выделить 5 сущностей: «Buildings(Здание)», «Rooms(Помещение)», «Room Parts(Составная часть помещения)», «Users(Пользователи)», «Decoration(Отделка)».

В приложении А на рисунке 3 представлена логическая модель данных информационной системы «Бюро технической инвентаризации».

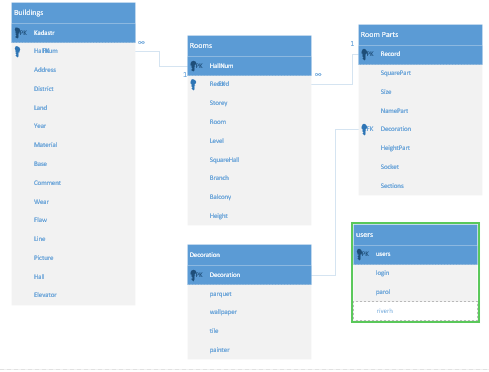


Рисунок 10 – Логическая модель данных

Создание базы данных происходит в MS SQL.

С помощью MS SQL были созданы все таблицы и поля, представленные на логической модели данных.

Названия полей и их типы данных представлены в таблицах 1-5.

Таблица 1 – Поля таблицы «Buildings»

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | | Тип данных | |
| Kadastr | | VARCHAR (50) | |
| HallNum | | INT | |
| Address | | INT | |
| District | | VARCHAR (50) | |
| Land | | VARCHAR (15) | |
| Year | | INT | |
| Materia | | VARCHAR (15) | |
| Base | | VARCHAR (15) | |
| Comment | | VARCHAR (MAx) | |
| Wear | | INT | |
| Flow | | INT | |
| Line | | INT | |
| Square | | INT | |
| Picture | | VARCHAR (50) | |
| Hall | | INT | |
| Elevator | | BIT | |

Таблица 2 – Поля таблицы «Rooms»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| HallNum | INT |
| Record | INT |
| Storey | INT |
| Rooms | INT |
| Level | BIT |
| Branch | FLOAT (53) |
| Balcony | FLOAT (53) |
| Height | FLOAT (53) |

Таблица 3 – Поля таблицы «Room Parts»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Record | INT |
| SquarePart | FLOAT (53) |
| Size | VARCHAR (40) |
| NamePart | VARCHAR (30) |
| Decoration | VARCHAR (60) |
| HeightPart | FLOAT (53) |
| Socket | INT |
| Sections | INT |

Таблица 4 – Поля таблицы «Users»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Users | INT IDENTITY (1, 1) |
| Login | VARCHAR (20) |
| Parol | VARCHAR (15) |
| Reverh | VARCHAR (15) |

Таблица 5 – Поля таблицы «Decoration»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Decoration | VARCHAR (60) |
| parquet | VARCHAR (20) |
| wallpaper | VARCHAR (20) |
| tile | VARCHAR (20) |
| painter | VARCHAR (20) |

На рисунке 11 представлены все созданные таблицы в обозревателе объектов базы данных.

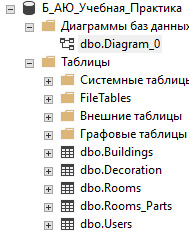


Рисунок 11 – Созданные таблицы

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

В приложении А на рисунке 4 представлена физическая модель данных, отражающая все таблицы и их поля, а также связи между таблицами.

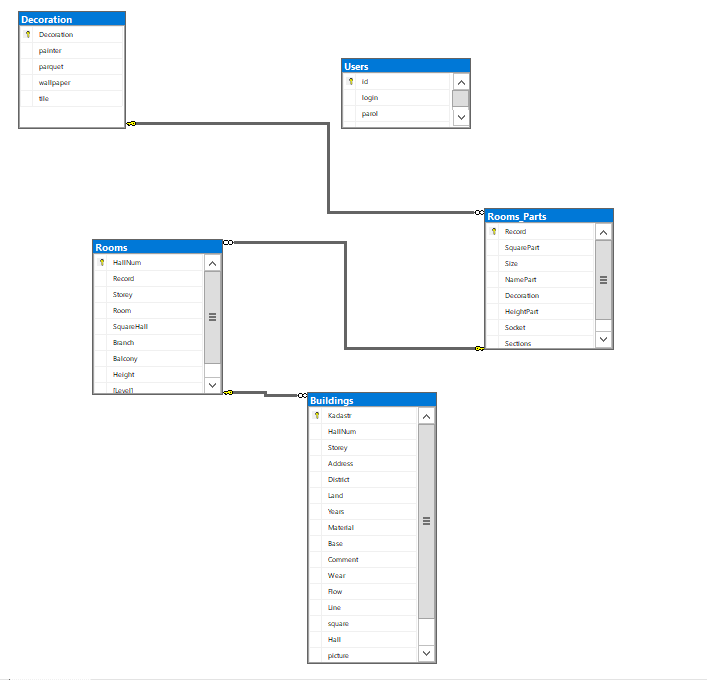


Рисунок 12 – Физическая модель базы данных

## Разработка программных модулей информационной системы «Бюро технической инвентаризации »

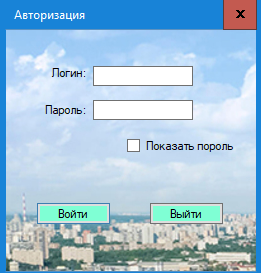


Рисунок 13 – Форма “Авторизация”

Свойства элементов формы entrance представлены в таблице 6.

Таблица 6 – Свойства элементов формы entrance

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| entrance | FormBorderStyle | SizableToolWindow |
| Text | Авторизация |
| StartPosition | CenterScreen |
| Size | 266; 280 |
| label1 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Логин: |
| ForeColor | Black |
| Label2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Пароль: |
| ForeColor | Black |
| Вход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Войти |
| ForeColor | Black |
| BackColor | Aquamarine |
| Выйти | Font | Microsoft Sans Serif; 8,25pt |
| BackColor | Aquamarine |
| Text | Выйти |
| ForeColor | Black |
| СкрытьПароль | Font | Microsoft Sans Serif; 8,25pt |
| Text | checkBox1 |
| ForeColor | ControlText |
|  |  |

Событийно-управляемые процедуры в форме авторизации представлены в таблице 7

Таблица 7 – Событийно-управляемые процедуры в форме «Авторизация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Вход\_Click\_1 | if ((textBox1.Text == "") || (textBox2.Text == ""))  {  if ((textBox1.Text == "") && (textBox2.Text == ""))  {  MessageBox.Show("Заполните все поля!!!");  }  else  {  if ((textBox1.Text == "") && (textBox2.Text == ""))  {  MessageBox.Show("Введите Логин и Пароль!!!");  }  else  {  if (textBox1.Text == "")  {  MessageBox.Show("Введите Логин!!!");  }  else  {  if (textBox2.Text == "")  {  MessageBox.Show("Введите Пароль!!!");  }  }  }  }  }  else  {  Boolean flag = false;  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  SqlDataAdapter da = new SqlDataAdapter("select \* from Users", sqlConnect);  DataTable dt = new DataTable();  da.Fill(dt);  for (int i = 0; i < dt.Rows.Count; i++)  {  if ((dt.Rows[i]["login"].ToString() == textBox1.Text) &&  (dt.Rows[i]["parol"].ToString() == textBox2.Text))  {  string роль = dt.Rows[i]["reverh"].ToString();  switch (роль)  {  case "Админ":  this.Hide();  admin AdmFrm = new admin();  AdmFrm.Show();  break;  case "Сотрудник":  this.Hide();  admin AdmFr = new admin();  AdmFr.Show();  AdmFr.ii("Сотрудник");  break;  case "Директор":  this.Hide();  admin AdmF = new admin();  AdmF.Show();  AdmF.ii("Директор");  break;  default:  MessageBox.Show("Неправильный логин или пароль.");  break;  }  break;  }  }  }  } |
| Выйти\_Click\_1 | private void Выйти\_Click\_1(object sender, EventArgs e)  {  Application.Exit();  } |
| выйти\_Click | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| СкрытьПароль\_CheckedChanged | private void СкрытьПароль\_CheckedChanged(object sender, EventArgs e)  {  if (СкрытьПароль.Checked == true)  {  СкрытьПароль.Text = "Скрыть пороль";  textBox2.UseSystemPasswordChar = false;  }  else  {  СкрытьПароль.Text = "Показать пороль";  textBox2.UseSystemPasswordChar = true;  }  } |
| entrance\_Load | private void entrance\_Load(object sender, EventArgs e)  {  textBox2.UseSystemPasswordChar = true;  СкрытьПароль.Text = "Показать пороль";  } |

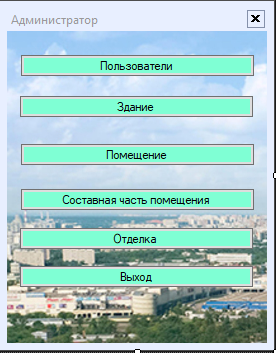


Рисунок 14 – Форма “ admin ”

Свойства элементов формы admin представлены в таблице 8.

Таблица 8 – Свойства элементов формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| admin | FormBorderStyle | SizableToolWindow |
| Text | Администратор |
| StartPosition | CenterScreen |
| Size | 276; 351 |
| Пользователи | Font | Microsoft Sans Serif; 8,25pt |
| Text | Пользователи |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Здание | Font | Microsoft Sans Serif; 8,25pt |
| Text | Здание |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Помещения | Font | Microsoft Sans Serif; 8,25pt |
| Text | Помещения |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| СоставнаяЧасть | Font | Microsoft Sans Serif; 8,25pt |
| Text | Составная часть помещения |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Отделка | Font | Microsoft Sans Serif; 8,25pt |
| Text | Отделка |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Выход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выход |
| ForeColor | ControlText |
| BackColor | Aquamarine |

Событийно-управляемые процедуры в форме admin представлены в таблице 9

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

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class admin | //переход по формам по правам доступа  string a;  public void ii(string b)  {  a = b;  i1();  }  public void i1()  {  if (a == "Сотрудник")  {  Text = "Панель Сотрудника";  Пользователи.Visible = false;  Здания.Location = new Point (12,23);  Помещения.Location = new Point(12, 73);  СоставнаяЧасть.Location = new Point(12, 121);  Отделка.Location = new Point(12, 157);  Выход.Location = new Point(12, 196);  }  if (a == "Админ")  {  Text = "Панель Администратора";  Пользователи.Visible = true;  }  if (a == "Директор")  {  Text = "Панель Директора";  }  } |
| Пользователи\_Click | private void Пользователи\_Click(object sender, EventArgs e)  {  //переход на форму  this.Hide();  Users AdmFrm = new Users();  AdmFrm.Show();  } |
| Здания\_Click | private void Здания\_Click(object sender, EventArgs e)  {  this.Hide();  buildings AdmFr = new buildings();  AdmFr.Show();  } |
| Помещения\_Click | private void Помещения\_Click(object sender, EventArgs e)  {  this.Hide();  Rooms AdmFr = new Rooms();  AdmFr.Show();  } |
| СоставнаяЧасть\_Click | private void СоставнаяЧасть\_Click(object sender, EventArgs e)  {  this.Hide();  Record AdmFr = new Record();  AdmFr.Show();  } |
| Отделка\_Click | private void Отделка\_Click(object sender, EventArgs e)  {  this.Hide();  decoration AdmFr = new decoration();  AdmFr.Show();  } |
| Выход\_Click | private void Выход\_Click(object sender, EventArgs e)  {  this.Hide();  entrance AdmFr = new entrance();  AdmFr.Show();  } |

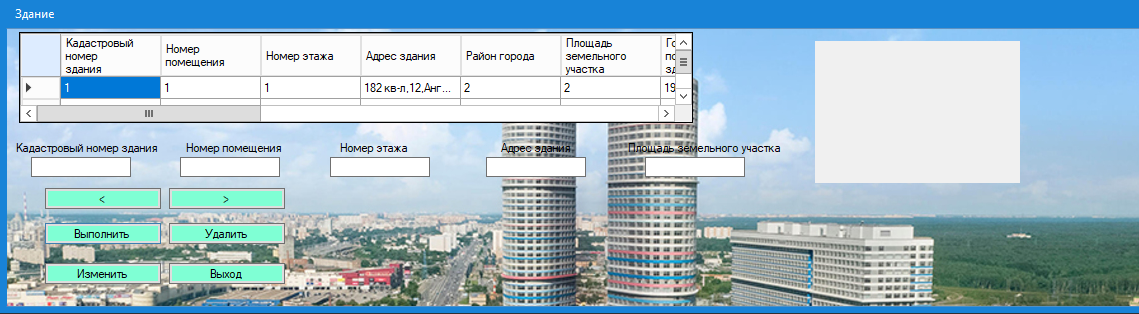


Рисунок 15 – Форма “ buildings ”

Свойства элементов формы buildings представлены в таблице 10.

Таблица 10 – Свойства элементов формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| buildings | FormBorderStyle | SizableToolWindow |
| Text | Здание |
| StartPosition | CenterScreen |
| Size | 1205; 316 |
| Переход | Font | Microsoft Sans Serif; 8,25pt |
| Text | > |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| переход2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | < |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Добавить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выполнить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Удалить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Удалить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Изменить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Изменить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Выход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выход |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| label1 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Кадастровый номер здания |
| ForeColor | Black |
| Label2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номерпомещения |
|  | ForeColor | Black |
| Label3 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номер этажа |
| ForeColor | Black |
| Label4 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Адрес здания |
|  | ForeColor | Black |
| Label5 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Площадь земельного участка |
| ForeColor | Black |
| Label6 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Количество помещений в здании |
| ForeColor | Black |
| textBox1 | Text | “” |
| textBox2 | Text | “” |
| textBox3 | Text | “” |
| textBox4 | Text | “” |
| textBox5 | Text | “” |
| textBox6 | Text | “” |
| textBox7 | Text | “” |
| textBox8 | Text | “” |
| textBox8 | Text | “” |
| textBox9 | Text | “” |
| textBox10 | Text | “” |
| textBox11 | Text | “” |
| textBox12 | Text | “” |
| textBox13 | Text | “” |
| textBox14 | Text | “” |
| textBox15 | Text | “” |

Событийно-управляемые процедуры в форме buildings представлены в таблице 11

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

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| buildings\_Load | // перевод название полей на русский  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter da = new SqlDataAdapter("select Kadastr as [Кадастровый номер здания], HallNum as [Номер помещения], Storey as [Номер этажа], Address as [Адрес здания], District as [Район города], Land as [Площадь земельного участка ], Years as [Год постройки здания], Material as [Материал стен здания], Base as [Материал фундамента], Comment as [Примечания], Wear as [Износ в процентах], Flow as [Количество этажей в здании], Line as [Расстояние от центра города], square as [Площадь нежилых помещений], picture as [Фото здания], Hall as [Количество помещений в здании], Elevator as [Наличие лифта] from Buildings", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  checkBox1.Visible = false;  textBox6.Visible = false;  textBox7.Visible = false;  textBox8.Visible = false;  textBox9.Visible = false;  textBox10.Visible = false;  textBox11.Visible = false;  textBox12.Visible = false;  textBox13.Visible = false;  textBox14.Visible = false;  textBox15.Visible = false;  Вставка.Visible = false; |
| Переход\_Click\_1 | //скрывание элементов формы и заплонение их через кнопку  private void Переход\_Click\_1(object sender, EventArgs e)  {  if (press == 0)  {  label2.Text = "Год постройки здания";  label3.Text = "Материал стен здания";  label4.Text = "Материал фундамента";  label5.Text = "Примечания";  label1.Text = "Площадь земельного участка";  textBox1.Visible = false;  textBox2.Visible = false;  textBox4.Visible = false;  textBox5.Visible = false;  textBox3.Visible = false;  textBox6.Visible = true;  textBox7.Visible = true;  textBox8.Visible = true;  textBox9.Visible = true;  textBox10.Visible = true;  press = 1;  label1.Location = new Point(6, 112);  label2.Location = new Point(176, 112);  label3.Location = new Point(330, 112);  label4.Location = new Point(491, 112);  label5.Location = new Point(650, 112);  }  else  {  if (press == 1)  {  label2.Text = "Количество этажей в здании";  label3.Text = "Расстояние от центра города";  label4.Text = "Площадь нежилых помещений";  label5.Text = "Фото здания";  label1.Text = "Год постройки здания";  textBox6.Visible = false;  textBox7.Visible = false;  textBox8.Visible = false;  textBox9.Visible = false;  textBox10.Visible = false;  textBox1.Visible = false;  textBox2.Visible = false;  textBox4.Visible = false;  textBox5.Visible = false;  textBox3.Visible = false;  textBox11.Visible = true;  textBox12.Visible = true;  textBox13.Visible = true;  textBox14.Visible = true;  textBox15.Visible = true;  checkBox1.Visible = true;  Вставка.Visible = true;  label6.Visible = true;  label2.Location = new Point(150, 112);  label3.Location = new Point(310, 112);  label4.Location = new Point(480, 112);  press = 0;  }  } |
| Добавить\_Click | // добавление данных в поля и обновление их  private void Добавить\_Click(object sender, EventArgs e)  {  int bit;  if (checkBox1.Checked == true)  {  bit = 1;  }  else bit = 0;  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"INSERT INTO Buildings (Kadastr,HallNum,Storey,Address,District,Land,Years,Material,Base,Comment,Wear,Flow,Line,square,Hall,Elevator, picture) VALUES ('{textBox1.Text}','{textBox2.Text}','{textBox3.Text}','{textBox4.Text}','{textBox5.Text}','{textBox6.Text}','{textBox7.Text}','{textBox8.Text}','{textBox9.Text}','{textBox10.Text}','{textBox11.Text}','{textBox12.Text}','{textBox13.Text}','{textBox14.Text}','{textBox15.Text}','{bit}','{pyt}');", sqlConnect);  da.Fill(dt);  MessageBox.Show("Данные внесены");  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Buildings", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  daa.Fill(dta);  } |
| Удалить\_Click | //удаление данных в полях  private void Удалить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"delete from Buildings where Kadastr = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Buildings", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } |
| Изменить\_Click | //изменения данных в полях  private void Изменить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  //,,,Room,,,,,),'{textBox6.Text},'')  SqlDataAdapter da = new SqlDataAdapter($"update Buildings set Kadastr = '{textBox1.Text}',HallNum = '{textBox2.Text}',Storey='{textBox3.Text}',Address='{textBox4.Text}',District='{textBox5.Text}',Land='{textBox6.Text}',Years='{textBox7.Text}',Material= '{textBox8.Text}',Base='{textBox9.Text}',Comment='{textBox10.Text}',Wear='{textBox11.Text}',Flow='{textBox12.Text}',Line='{textBox13.Text}',square='{textBox14.Text}',Hall='{textBox15.Text}' where Kadastr = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Buildings", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } |
| Вставка\_Click | // вставка пути картинки в поле  private void Вставка\_Click(object sender, EventArgs e)  {  if (openFileDialog1.ShowDialog() == DialogResult.OK)  {  pyt = openFileDialog1.FileName;  pictureBox1.Image = Image.FromFile(pyt);  }  } |
| Выход\_Click | private void Выход\_Click(object sender, EventArgs e)  {  this.Hide();  admin AdmFr = new admin();  AdmFr.Show(); } |
| переход2\_Click | //скрывание элементов формы и заплонение их через кнопку  if (press == 0)  {  label2.Text = "Количество этажей в здании";  label3.Text = "Расстояние от центра города";  label4.Text = "Площадь нежилых помещений";  label5.Text = "Фото здания";  label1.Text = "Год постройки здания"; ;  textBox1.Visible = false;  textBox2.Visible = false;  textBox4.Visible = false;  textBox5.Visible = false;  textBox3.Visible = false;  textBox6.Visible = true;  textBox7.Visible = true;  textBox8.Visible = true;  textBox9.Visible = true;  textBox10.Visible = true;  press = 1;  label1.Location = new Point(6, 112);  label2.Location = new Point(176, 112);  label3.Location = new Point(330, 112);  label4.Location = new Point(491, 112);  label5.Location = new Point(650, 112);  Вставка.Visible = false;  checkBox1.Visible = false;  label6.Visible = false;  }  else  {  if (press == 1)  {  label2.Text = "Номер помещения";  label3.Text = "Номер этажа";  label4.Text = "Адрес здания";  label5.Text = "Площадь земельного участка";  label1.Text = "Кадастровый номер здания";  textBox6.Visible = false;  textBox7.Visible = false;  textBox8.Visible = false;  textBox9.Visible = false;  textBox10.Visible = false;  textBox1.Visible = true;  textBox2.Visible = true;  textBox4.Visible = true;  textBox5.Visible = true;  textBox3.Visible = true;  textBox11.Visible = false;  textBox12.Visible = false;  textBox13.Visible = false;  textBox14.Visible = false;  textBox15.Visible = false;  checkBox1.Visible = false;  Вставка.Visible = false;  label6.Visible = false;  label1.Location = new Point(6, 112);  label2.Location = new Point(176, 112);  label3.Location = new Point(330, 112);  label4.Location = new Point(491, 112);  label5.Location = new Point(618, 112);  press = 0;  }  } |
| checkBox1\_CheckedChanged | // заполнения поля через checbox  int bit = 0;  if (checkBox1.Checked == true)  {  bit = 1;  }  else bit = 0;  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataTable dta = new DataTable();  SqlDataAdapter daa = new SqlDataAdapter($"INSERT INTO Buildings(Elevator) VALUES ('{bit}');", sqlConnect);    } |
| dataGridView1\_CellClick | //внос пути картинки в поле  pictureBox1.Image = Image.FromFile(dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[16].Value.ToString()); |

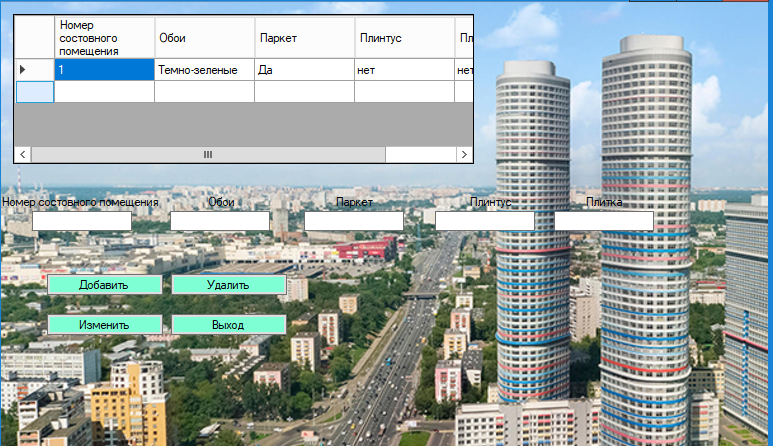


Рисунок 16 – Форма “ decoration ”

Свойства элементов формы decoration представлены в таблице 12.

Таблица 12 – Свойства элементов формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| decoration | FormBorderStyle | SizableToolWindow |
| Text | Отделка |
| StartPosition | CenterScreen |
| Size | 783; 489 |
| Добавить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выполнить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Удалить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Удалить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Изменить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Изменить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Выход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выход |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| label1 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номер состовного помещения |
| ForeColor | Black |
| Label2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Обои |
|  | ForeColor | Black |
| Label3 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Паркет |
| ForeColor | Black |
| Label4 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Плинтус |
|  | ForeColor | Black |
| Label5 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Плитка |
| ForeColor | Black |
| textBox1 | Text | “” |
| textBox2 | Text | “” |
| textBox3 | Text | “” |
| textBox4 | Text | “” |
| textBox5 | Text | “” |

Событийно-управляемые процедуры в форме decoration представлены в таблице 13

Таблица 13 – Событийно-управляемые процедуры в форме «decoration»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| decoration\_Load | private void decoration\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "б\_АЮ\_Учебная\_ПрактикаDataSet2.Decoration". При необходимости она может быть перемещена или удалена.  this.decorationTableAdapter.Fill(this.б\_АЮ\_Учебная\_ПрактикаDataSet2.Decoration);  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter da = new SqlDataAdapter("select Decoration as [Номер состовного помещения], painter as [Обои], parquet as [Паркет], wallpaper as [Плинтус],tile as [Плитка] from Decoration", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  } |
| Добавить\_Click | private void Добавить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"INSERT INTO Decoration (Decoration,painter,parquet,wallpaper,tile) VALUES ('{textBox1.Text}','{textBox2.Text}','{textBox3.Text}','{textBox4.Text}','{textBox5.Text}');", sqlConnect);  da.Fill(dt);  decorationTableAdapter.Update(б\_АЮ\_Учебная\_ПрактикаDataSet2.Decoration);  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Decoration", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  MessageBox.Show("Данные добавленны");  } |
| удалить\_Click | private void удалить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"delete from Decoration where Decoration = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Decoration", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } |
| Изменить\_Click | private void Изменить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  //,,,Room,,,,,),'{textBox6.Text},'')  SqlDataAdapter da = new SqlDataAdapter($"update Decoration set Decoration = '{textBox1.Text}',painter = '{textBox2.Text}',parquet='{textBox3.Text}',wallpaper='{textBox4.Text}',tile='{textBox5.Text}' where Decoration = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Decoration", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } |
| Выход\_Click | private void Выход\_Click(object sender, EventArgs e)  {  this.Hide();  admin AdmFr = new admin();  AdmFr.Show();  } |

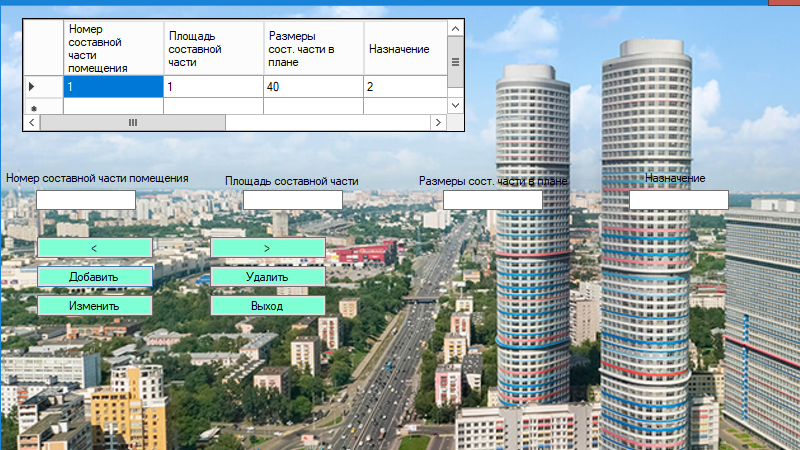


Рисунок 17 – Форма “ Record ”

Свойства элементов формы Record представлены в таблице 14.

Таблица 14 – Свойства элементов формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Record | FormBorderStyle | SizableToolWindow |
| Text | Отделка |
| StartPosition | CenterScreen |
| Size | 816; 489 |
| Добавить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выполнить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Удалить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Удалить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Изменить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Изменить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Выход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выход |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| label1 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номер составной части помещения |
| ForeColor | Black |
| Label2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Площадь составной части |
|  | ForeColor | Black |
| Label3 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Размеры сост. части в плане |
| ForeColor | Black |
| Label4 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Назначение |
|  | ForeColor | Black |
| textBox1 | Text | “” |
| textBox2 | Text | “” |
| textBox3 | Text | “” |
| textBox4 | Text | “” |

Событийно-управляемые процедуры в форме Record представлены в таблице 15

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

| Объект | Событийно-управляемые процедуры | |
| --- | --- | --- |
| Record\_Load | private void Record\_Load(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter da = new SqlDataAdapter("select Record as [Номер составной части помещения], SquarePart as [Площадь составной части], Size as [Размеры сост. части в плане], NamePart as [Назначение],Decoration as [Отделка], HeightPart as [Высота составной части], Socket as [Число розеток в помещении], Sections as [Число элементов в батарее отопления] from Rooms\_Parts", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  textBox5.Visible = false;  textBox6.Visible = false;  textBox7.Visible = false;  textBox8.Visible = false;  } | |
| Добавить\_Click | private void Добавить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"INSERT INTO Rooms\_Parts (Record,SquarePart,Size,Room,NamePart,Decoration,Socket,Sections) VALUES ('{textBox1.Text}','{textBox2.Text}','{textBox3.Text}','{textBox4.Text}','{textBox5.Text}','{textBox6.Text}','{textBox7.Text}','{textBox8.Text}');", sqlConnect);  da.Fill(dt);  MessageBox.Show("Данные внесены");  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| удалить\_Click | private void удалить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"delete from Rooms\_Parts where Record = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms\_Parts", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| Изменить\_Click | private void Изменить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"update Rooms\_Parts set Record = '{textBox1.Text}',SquarePart = '{textBox2.Text}',Size='{textBox3.Text}',NamePart='{textBox4.Text}',Decoration='{textBox5.Text}',HeightPart='{textBox6.Text}',Socket='{textBox7.Text}',Sections= '{textBox8.Text}' where Record = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| Выход\_Click | | private void Выход\_Click(object sender, EventArgs e)  {  this.Hide();  admin AdmFr = new admin();  AdmFr.Show();  } |
| Переход\_Click | | private void Переход\_Click(object sender, EventArgs e)  {  textBox1.Visible = false;  textBox2.Visible = false;  textBox3.Visible = false;  textBox4.Visible = false;  textBox5.Visible = true;  textBox6.Visible = true;  textBox7.Visible = true;  textBox8.Visible = true;  label1.Text = "Отделка";  label2.Text = "Высота составной части";  label3.Text = "Число розеток в помещении";  label4.Text = "Число элементов в батарее отопления";  label1.Location = new Point(41, 165);  label4.Location = new Point(590, 165);  } |
| Переход2\_Click | | textBox8.Visible = false;  textBox7.Visible = false;  textBox6.Visible = false;  textBox5.Visible = false;  textBox1.Visible = true;  textBox2.Visible = true;  textBox3.Visible = true;  textBox4.Visible = true;  label1.Text = "Номер составной части помещения";  label2.Text = "Площадь составной части";  label3.Text = "Размеры сост. части в плане";  label4.Text = "Назначение";  label1.Location = new Point(2, 165);  label4.Location = new Point(641, 165); |

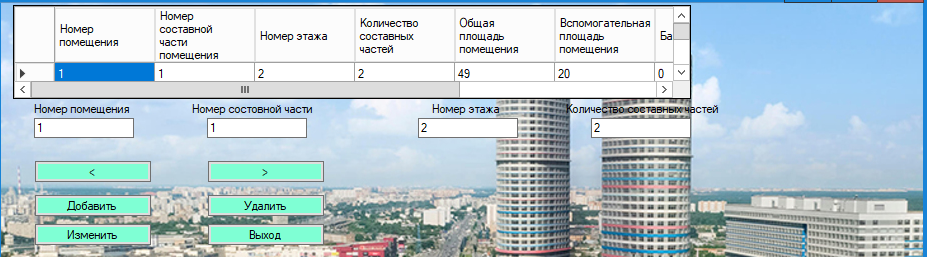


Рисунок 18 – Форма “ Rooms ”

Свойства элементов формы Rooms представлены в таблице 16.

Таблица 16 – Свойства элементов формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Rooms | FormBorderStyle | SizableToolWindow |
| Text | Помещение |
| StartPosition | CenterScreen |
| Size | 938; 294 |
| Добавить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Добавить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Удалить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Удалить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Изменить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Изменить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Выход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выход |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| label1 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номер помещения |
| ForeColor | Black |
| Label2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номер состовной части |
|  | ForeColor | Black |
| Label3 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Номер этажа |
| ForeColor | Black |
| Label4 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Количество составных частей |
|  | ForeColor | Black |
| textBox1 | Text | “” |
| textBox2 | Text | “” |
| textBox3 | Text | “” |
| textBox4 | Text | “” |
| textBox5 | Text | “” |
| textBox6 | Text | “” |
| textBox7 | Text | “” |
| textBox8 | Text | “” |

Событийно-управляемые процедуры в форме Rooms представлены в таблице 17

Таблица 17 – Событийно-управляемые процедуры в форме «Rooms»

| Объект | Событийно-управляемые процедуры | |
| --- | --- | --- |
| Rooms\_Load | private void Rooms\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "б\_АЮ\_Учебная\_ПрактикаDataSet.Buildings". При необходимости она может быть перемещена или удалена.  //this.buildingsTableAdapter.Fill(this.б\_АЮ\_Учебная\_ПрактикаDataSet.Buildings);  // TODO: данная строка кода позволяет загрузить данные в таблицу "б\_АЮ\_Учебная\_ПрактикаDataSet.Rooms". При необходимости она может быть перемещена или удалена.  this.roomsTableAdapter.Fill(this.б\_АЮ\_Учебная\_ПрактикаDataSet.Rooms);  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter da = new SqlDataAdapter("select HallNum as [Номер помещения], Record as [Номер составной части помещения], Storey as [Номер этажа], Room as [Количество составных частей],SquareHall as [Общая площадь помещения], Branch as [Вспомогательная площадь помещения], Balcony as [Балкон], Height as [Высота помещения], Level as [Помещение в двух уровнях] from Rooms", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  } | |
| Добавить\_Click | private void добавить\_Click(object sender, EventArgs e)  {  int bit = 0;  if (checkBox1.Checked == true)  {  bit = 1;  }  else bit = 0;  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"INSERT INTO Rooms (HallNum,Record,Storey,Room,SquareHall,Branch,Balcony,Height,Level) VALUES ('{textBox1.Text}','{textBox2.Text}','{textBox3.Text}','{textBox4.Text}','{textBox5.Text}','{textBox6.Text}','{textBox7.Text}','{textBox8.Text}','{bit}');", sqlConnect);  da.Fill(dt);  //dataGridView2.DataSource = dt.Tables[0];  roomsTableAdapter.Update(б\_АЮ\_Учебная\_ПрактикаDataSet.Rooms);  MessageBox.Show("Данные внесены");  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| удалить\_Click | private void удалить\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  SqlDataAdapter da = new SqlDataAdapter($"delete from Rooms where HallNum = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| Изменить\_Click | private void изменить\_Click(object sender, EventArgs e)  {  int bit = 0;  if (checkBox1.Checked == true)  {  bit = 1;  }  else bit = 0;  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  //,,,Room,,,,,),'{textBox6.Text},'')  SqlDataAdapter da = new SqlDataAdapter($"update Rooms set Room = '{textBox4.Text}',HallNum = '{textBox1.Text}',Record='{textBox2.Text}',Storey='{textBox3.Text}',SquareHall='{textBox5.Text}',Branch='{textBox6.Text}',Balcony='{textBox7.Text}',Height= '{textBox8.Text}',Level='{bit}' where HallNum = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| Выход\_Click | | private void Выход\_Click(object sender, EventArgs e)  {  this.Hide();  admin AdmFr = new admin();  AdmFr.Show();  } |
| Переход\_Click | | private void Переход\_Click(object sender, EventArgs e)  {  label1.Text = "Общая площадь помещения";  label2.Text = "Вспомогательная площадь помещения";  label3.Text = "Балкон";  label4.Text = "Высота помещения";  textBox1.Visible = false;  textBox2.Visible = false;  textBox3.Visible = false;  textBox4.Visible = false;  textBox5.Visible = true;  textBox6.Visible = true;  textBox7.Visible = true;  textBox8.Visible = true;  checkBox1.Visible = true;  } |
| Переход2\_Click | | private void Переход2\_Click(object sender, EventArgs e)  {  label1.Text = "Номер помещения";  label2.Text = "Номер состовной части";  label3.Text = "Номер этажа";  label4.Text = "Количество составных частей";  textBox1.Visible = true;  textBox2.Visible = true;  textBox3.Visible = true;  textBox4.Visible = true;  textBox5.Visible = false;  textBox6.Visible = false;  textBox7.Visible = false;  textBox8.Visible = false;  checkBox1.Visible = false;  } |

Свойства элементов формы Rooms представлены в таблице 18.

Таблица 18 – Свойства элементов формы

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Users | FormBorderStyle | SizableToolWindow |
| Text | Пользователи |
| StartPosition | CenterScreen |
| Size | 938; 294 |
| Добавить | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выполнить |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Найти | Font | Microsoft Sans Serif; 8,25pt |
| Text | Найти |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| Выход | Font | Microsoft Sans Serif; 8,25pt |
| Text | Выход |
| ForeColor | ControlText |
| BackColor | Aquamarine |
| label1 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Логин: |
| ForeColor | Black |
| Label2 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Пароль: |
|  | ForeColor | Black |
| Label3 | Font | Microsoft Sans Serif; 8,25pt |
| Text | Право доступа: |
| ForeColor | Black |
| textBox1 | Text | “” |
| textBox2 | Text | “” |
| textBox3 | Text | “” |
| textBox4 | Text | “” |

Событийно-управляемые процедуры в форме Rooms представлены в таблице 19

Таблица 19 – Событийно-управляемые процедуры в форме «Rooms»

| Объект | Событийно-управляемые процедуры | |
| --- | --- | --- |
| Users\_Load | private void Users\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "б\_АЮ\_Учебная\_ПрактикаDataSet1.Users". При необходимости она может быть перемещена или удалена.  this.usersTableAdapter1.Fill(this.б\_АЮ\_Учебная\_ПрактикаDataSet1.Users);  // TODO: данная строка кода позволяет загрузить данные в таблицу "б\_АЮ\_Учебная\_ПрактикаDataSet.Users". При необходимости она может быть перемещена или удалена.  //this.usersTableAdapter.Fill(this.б\_АЮ\_Учебная\_ПрактикаDataSet.Users);  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter da = new SqlDataAdapter("select login as [логин], parol as [пароль], reverh as [Права доступа] from Users", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  } | |
| Добавить\_Click | SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataTable dta = new DataTable();  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Users", sqlConnect);  daa.Fill(dta);  string n = dta.Rows[dta.Rows.Count - 1]["id"].ToString();  SqlDataAdapter da = new SqlDataAdapter($"INSERT INTO Users (login,parol,reverh) VALUES ('{textBox1.Text}','{textBox2.Text}','{textBox3.Text}');", sqlConnect);  da.Fill(dt);  usersTableAdapter.Update(б\_АЮ\_Учебная\_ПрактикаDataSet.Users);  MessageBox.Show($"Регистрация пользователя с логином {textBox1.Text} выполнена.\n"); | |
| Найти\_Click | private void Найти\_Click(object sender, EventArgs e)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Users where login like '{textBox4.Text}%' ", sqlConnect);  daa.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  } | |
| Изменить\_Click | private void изменить\_Click(object sender, EventArgs e)  {  int bit = 0;  if (checkBox1.Checked == true)  {  bit = 1;  }  else bit = 0;  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  DataSet dta = new DataSet();  //,,,Room,,,,,),'{textBox6.Text},'')  SqlDataAdapter da = new SqlDataAdapter($"update Rooms set Room = '{textBox4.Text}',HallNum = '{textBox1.Text}',Record='{textBox2.Text}',Storey='{textBox3.Text}',SquareHall='{textBox5.Text}',Branch='{textBox6.Text}',Balcony='{textBox7.Text}',Height= '{textBox8.Text}',Level='{bit}' where HallNum = {dataGridView1.Rows[dataGridView1.SelectedCells[0].RowIndex].Cells[0].Value.ToString()} ;", sqlConnect);  da.Fill(dt);  dataGridView1.DataSource = dt;  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Rooms", sqlConnect);  daa.Fill(dta);  dataGridView1.DataSource = dta.Tables[0];  } | |
| Выйти\_Click | | private void Выход\_Click(object sender, EventArgs e)  {  this.Hide();  admin AdmFr = new admin();  AdmFr.Show();  } |
| textBox4\_KeyPress | | private void textBox4\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar==(char)Keys.Enter)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = Б\_АЮ\_Учебная\_Практика; Integrated Security = True");  sqlConnect.Open();  DataSet dt = new DataSet();  SqlDataAdapter daa = new SqlDataAdapter($"select \* from Users where login like '{textBox4.Text}%' ", sqlConnect);  daa.Fill(dt);  dataGridView1.DataSource = dt.Tables[0];  }  } |

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

# 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. Тестовый сценарий может быть написан вручную (для выполнения человеком) либо полностью или частично автоматизирован.

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

Можно определить такие основные цели тестирования программного обеспечения:

* Предоставление информации о качестве ПО конечному заказчику.
* Повышение качества ПО.
* Предотвращение появления дефектов.

Цели тестирования могут отличаться, в зависимости от этапа разработки программного обеспечения, на котором оно проводится. К примеру, на этапе кодирования целью тестирования будет вызов как можно большего количества сбоев в работе программы, что позволит локализовать и исправить дефекты. В то же время, при приемочном тестировании необходимо показать, что система работает правильно. В период сопровождения, тестирование в основном необходимо для того, чтобы убедится в отсутствии новых багов, появившихся во время внесения изменений.

Главная же задача тестирования – поиск дефектов.

В данном разделе проведены два вида тестирования. Провести анализ тестирования и выбрать, как будет проходить тестирование ИС.

Модульное тестирование Цель: проверить, что код работает именно так, как должен (при заданных входных параметрах выдает предполагаемый результат).

Анализ покрытия кода тестами. Цель: проверить, что весь наш код отрабатывает при модульном тестировании, что нет не участвующих в тестировании участков кода.

Анализ стилистики кода. Цель: проверка кода на соблюдение стандартов разработки (отраслевых, корпоративных, принятых на проекте).

Анализ производительности. Цель: проанализировать производительность ключевых операций, потребление памяти, утечку памяти.

## 5.2 Технология тестирования

Рассмотрим, какие технологии можно применить для модульного тестирования и покрытия кода тестами.

Как в ходе выполнения проекта, так и при развитии системы, изменения в ее алгоритмы вносятся в соответствии с техническими заданиями, хотя данный документ может иметь различные названия. В технических заданиях, как правило, описываются критерии приемки: как мы и заказчик убедимся в том, что заказанный функционал работает верно.

В качестве примера можно привести такой упрощенный критерий:

* Исходные данные: в систему внесены данные о здании, помещениях внутри и отделки.
* В результате все данные сохраняются и их можно просмотреть.
* Вручную такое тестирование как правило выполняется единожды, поскольку выполнять каждый раз полный комплект тестов очень трудоемко.
* Нет гарантии того, что тесты, описанные в техническом задании, покрывают функциональность полностью. Т.е. даже если тесты описаны и проводятся регулярно, вероятно наличие не протестированного функционала.

Автоматизированное тестирование лишено данных недостатков. Конечно, возникает отдельная задача кодирования данного теста на основании критериев приемки, но это скорее дисциплинирующий фактор…

Прежде чем начинать юзабилити-тестирование необходимо собрать группу пользователей, которые будут тестировать данную информационную систему. Количество привлеченных пользователей должно быть не менее пяти человек. Пользователям будет представлен список некоторых вопросов, относящихся к графическому интерфейсу.

В таблице 20 представлены сведения о пользователях.

Таблица 20 – Информация о пользователях

|  |  |  |  |
| --- | --- | --- | --- |
| ФИО | Пол | Возраст | Образование |
| Буйлов Михаил Алексеевич | Мужской | 19 | Среднее специальное |
| Попов Семен Евгеньевич | Мужской | 20 | Среднее специальное |
| Неверов Дима Сергеевич | Мужской | 18 | Среднее специальное |
| Швилко Иван Анатольевич | Мужской | 18 | Среднее специальное |
| Степурин Никита Сергеевич | Мужской | 20 | Среднее образование |

## Результаты проведения тестирования

В ходе тестирования был применен метод нагрузочного тестирования и юзабилити-тестирования, которые проверяют работу при нагрузке системы и сделать простым и удобным в пользовании информационной системы.

Суть тестирования заключается в проверке работы всех обязательных функций.

Результаты проведения нагрузочного тестирования

Для проведения нагрузочного тестирования составлена таблица 28, в которой указано количество запущенных копий программного продукта, нагрузка на процесс и оперативную память.

Таблица 21 – Результаты нагрузочного тестирования

|  |  |  |
| --- | --- | --- |
| Число копий | Нагрузка на процессор(%) | Нагрузка на память(%) |
| 1 | 0,1 | 0,1 |
| 5 | 0,8 | 0,6 |
| 10 | 4,8 | 4,5 |
| 15 | 10,5 | 9 |
| 20 | 14,7 | 11,5 |

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.

Таблица 22 – Результаты проведения юзабилити-тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вопрос | Степурин Никита Сергеевич | Швилко Иван Анатольевич | Неверов Дима Сергеевич | Попов Семен  Евгеньевич | Буйлов Михаил Алексеевич |
| Присутствуют ли такие дизайн-элементы, которые неприятны для зрения? | нет | нет | нет | нет | нет |
| Присутствует ли отвлекающая информация? | нет | нет | нет | нет | нет |
| Понятна ли структура приложения на интуитивном уровне? | да | нет | да | да | да |
| Смогли ли вы найти интересующую вас информацию без затруднений? | да | да | да | нет | да |
| Понятен ли смысл приложения? | да | да | да | нет | да |
| Общая оценка | 10/10 | 8/10 | 8/10 | 8/10 | 10/10 |

Исходя из средней оценки информационной системы, рассчитанной на основе общих оценок всех пользователей с учетом ответов на все поставленные вопросы, можно сделать вывод о том, что внешние характеристики графического интерфейса находятся на довольно высоком уровне. Также в информационной системе нет помех для пользователя в виде отвлекающей информации, так как все отображаемые данные находятся в рамках предметной области.

Вывод по главе: были проведены тесты путём проведения модульного тестирования, анализа покрытия кода тестами, анализа стилистики кода и анализом производительности. Также было проведено нагрузочное тестирование, в результате чего нагрузка на процессор и память не имела критичных значений.

# 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, а клиентская часть в MS Visual Studio.

В MS SQL была создана база данных для хранения информации о зданиях, помещений, составных помещений, и отделки.

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 поиск по таблицам.

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 создания возможности расширить форму на полный экран или улучшения и изменения дизайна.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. IT-GOST.RU [Электронный ресурс]: Теория и практика UML. Диаграмма деятельности, 2023. URL: <http://it-gost.ru/articles/view_articles/96> (дата обращения: 20.11.2023).
2. [Электронный ресурс]: Программа профессионального модуля ПМ 01. Разработка модулей программного обеспечения для компьютерных систем,2019. URL: <https://cs.гапоу-кгпт.рф/-/dpBw6QvcMrs97SUYUOCWug/sv/document/e2/66/37/682986/2747/РП%20ПМ%2001.pdf?1680595748> (дата обращения: 18.11.2023).
3. ГОСТ 20886-85. Организация данных в системах обработки данных. Термины и определения [Электронный доступ]. — Режим доступа: [ГОСТ 20886-85. Организация данных в системах обработки данных. Термины и определения (rostest.info)](https://rostest.info/gost/001.001.040.035/gost-20886-85/#:~:text=%D0%93%D0%9E%D0%A1%D0%A2%2020886%2D85.%20%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F%20%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%20%D0%B2,%D0%BD%D0%B0%D1%83%D1%87%D0%BD%D0%BE%2D%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D1%87%D0%B5%D1%81%D0%) (Дата обращения: 01.12.2023 г.)
4. ГОСТ Р. 59793–2021. Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания [Электронный доступ]. — Режим доступа: [ГОСТ Р. 59793-2021. Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания (internet-law.ru)](https://internet-law.ru/gosts/gost/77858/?ysclid=lifap277mg519765664) (Дата обращения: 01.12.2023 г.).
5. ГОСТ 19.201-78. ЕСПД. Общие положения. [Электронный доступ]. — Режим доступа: [ГОСТ 19.201-78.ЕСПД. Техническое задание. Требования к содержанию и оформлению (swrit.ru)](https://www.swrit.ru/doc/espd/19.201-78.pdf?ysclid=lpi5u6naom706376064) (Дата обращения: 01.12.2023 г.).
6. Microsoft Visual Studio [Электронный доступ]. — Режим доступа: [Руководство по разработке модулей расширений на C# — Павел Еремеев (pvs-studio.ru)](https://pvs-studio.ru/ru/blog/posts/csharp/a0082/) (Дата обращения: 01.12.2023 г.).
7. Microsoft Visual Studios [Электронный доступ]. — Режим доступа: [Microsoft Visual Studios — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio) (Дата обращения: 01.12.2023 г.).
8. Microsoft SQL Server [Электронный доступ]. — Режим доступа: [Microsoft SQL Server — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/Microsoft_SQL_Server) (Дата обращения: 01.12.2023 г.).
9. Энтони Молинаро. SQL. Сборник рецептов — М: «БХВ-Петербург», 2017. — 591 с.
10. Кариев Ч. А. Разработка Windows-приложений на основе Visual C# (+ CD-ROM) — М: Интернет-университет информационных технологий, Бином. Лаборатория знаний, 2018. — 768 c.
11. Microsoft SQL Server [Электронный доступ]. — Режим доступа: [Как устроен язык SQL и почему он так востребован — Анализ данных (practicum.yandex.ru)](https://practicum.yandex.ru/blog/chto-takoe-sql/) (Дата обращения: 27.07.2022 г.).

# ПРИЛОЖЕНИЕ A

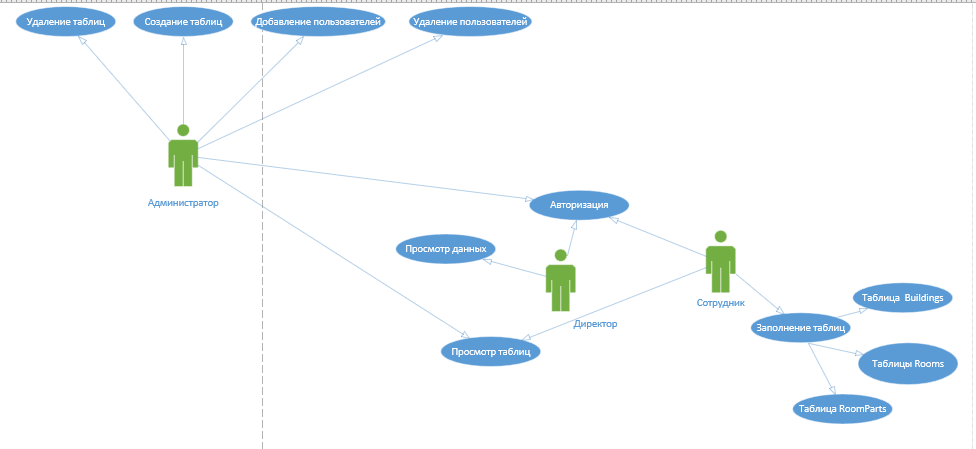


Рисунок 1 – Диаграмма вариантов использования

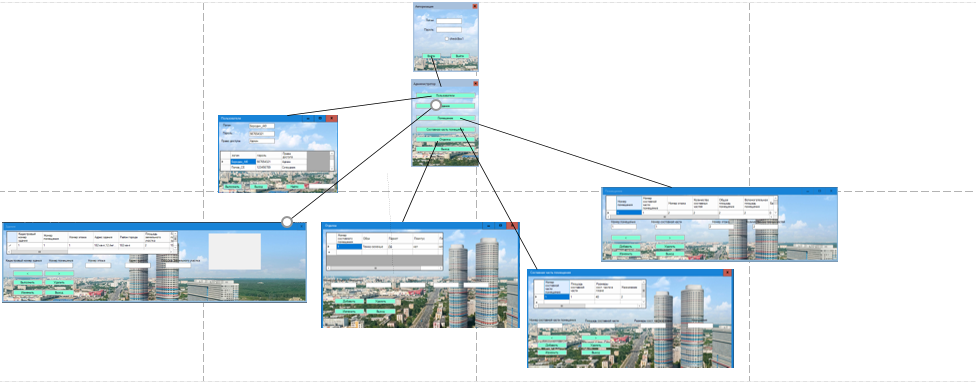


Рисунок 2 – Карта переходов

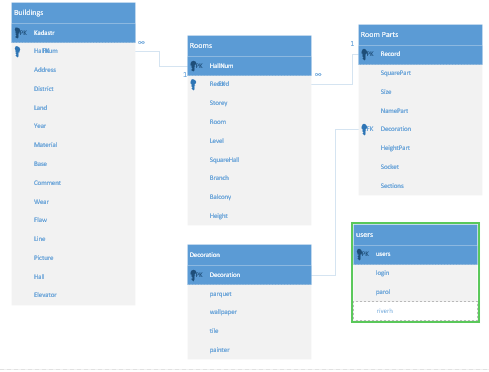


Рисунок 3 – Логическая модель данных

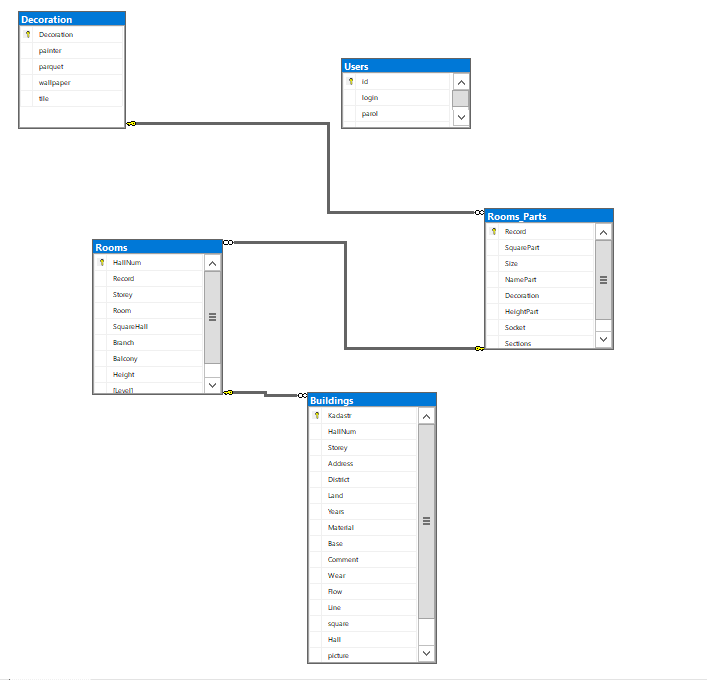


Рисунок 4 – Физическая модель базы данных