Содержание

[Введение 4](#_Toc151737033)

[1 Цель и задачи УЧЕБНОЙ практики. 5](#_Toc151737034)

[1.1 Основные требования по технике безопасности во время прохождения учебной практики 6](#_Toc151737035)

[1.2 Нормативно-правовые документы 7](#_Toc151737036)

[2 Предметная область. Анализ предметной области. Постановка задачи. 9](#_Toc151737037)

[2.1 Описание предметной области 9](#_Toc151737038)

[2.2 Группы пользователей информационной системы 9](#_Toc151737039)

[2.3 Основные требования, предъявляемые к информационной системе 10](#_Toc151737040)

[3 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «Охрана квартир» 11](#_Toc151737041)

[3.1 Моделирование некоторых динамических аспектов системы 11](#_Toc151737042)

[3.1.1 Создание диаграммы вариантов использования 12](#_Toc151737043)

[3.1.2 Создание диаграммы деятельности 14](#_Toc151737044)

[3.2 Разработка функциональной структуры информационной системы 15](#_Toc151737045)

[3.3 Разработка макета информационной системы 15](#_Toc151737046)

[4 РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ «Охрана квартир» 16](#_Toc151737047)

[4.1 Моделирование и разработка базы данных 16](#_Toc151737048)

[4.2. Разработка программных модулей информационной системы «Охрана квартир» 21](#_Toc151737049)

[5 РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ 22](#_Toc151737050)

[5.1 Назначение эксперимента. Выбор и обоснование методики проведения тестирования 22](#_Toc151737051)

[5.2 Технология тестирования 23](#_Toc151737052)

[5.3 Результаты проведения тестирования 23](#_Toc151737053)

[ЗАКЛЮЧЕНИЕ 24](#_Toc151737054)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 25](#_Toc151737055)

[ПРИЛОЖЕНИЕ A 26](#_Toc151737056)

[ПРИЛОЖЕНИЕ Б 30](#_Toc151737057)

# Введение

Каждому человеку нужна охрана своего имущества и своей квартиры. Современные вызовы в области безопасности жилья подчеркивает необходимость эффективных методов и технологий для защиты жилищных помещений. Один из таких подходов — разработка прикладного программного обеспечения для отделов вневедомственной охраны квартир. Это программное обеспечение будет предоставлять инновационные решения, направленные на повышение эффективности мер по обеспечению безопасности от злоумышленников. Разработка прикладного программного обеспечения для отдела вневедомственной охраны квартир представляет собой важный шаг в направлении повышения общей безопасности общества.

Для освоения основного вида профессиональной деятельности: «Разработка программного обеспечения для компьютерных систем» и соответствующих профессиональных компетенций:

ПК 1.1 - Выполнять разработку спецификаций отдельных компонент

ПК 1.2 - Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля

ПК 1.3 - Выполнять отладку программных модулей с использованием специализированных программных средств

ПК 1.4 - Выполнять тестирование программных модулей

ПК 1.5 - Осуществлять оптимизацию программного модуля

ПК 1.6 - Разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций

# Цель и задачи УЧЕБНОЙ практики.

Целью учебной практики является овладение видом профессиональной деятельности Разработка модулей программного обеспечения для компьютерных систем, общими и профессиональными компетенциями по специальности.

Основные задачи учебной практики:

* систематизация, обобщение, закрепление и углубление знаний и умений;
* формирование у студентов знаний, умений и навыков, профессиональных компетенций, профессионально значимых личностных качеств;
* развитие профессионального интереса, формирование мотивационно-целостного отношения к профессиональной деятельности, готовности к выполнению профессиональных задач в соответствии с нормами морали, профессиональной этики и служебного этикета.

Учебная практика профессионального модуля ПМ 01 Разработка модулей программного обеспечения для компьютерных систем проводится в кабинетах информационно - вычислительного центра техникума преподавателями общепрофессиональных дисциплин профессионального цикла.

В период выполнения заданий учебной практики профессиональных модулей на студентов распространяются требования охраны труда и правила внутреннего распорядка, действующие в ГБПОУ ИО «АПЭТ»

Электронная охрана квартир позволяет добиться безопасности от злоумышленников в районе. Программный продукт представляет собой универсальное программное обеспечение, предназначенное для деятельности отдела вневедомственной охраны квартир, для удобной эксплуатации в целях предоставления информации.

Программный продукт предназначен для индивидуального применения, то есть предполагает использование одним пользователем в целях получения информации. В программном продукте можно будет заключать договор на охрану квартир.

## 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 про1раммы 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.

# 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

В рамках учебной практики было выдано следующее задание: разработать прикладное программное обеспечение деятельности отдела вневедомственной охраны квартир. Этот отдел обеспечивает электронную охрану квартир граждан в одном районе города. Для установки охранной сигнализации требуется наличие квартирного телефона. Один гражданин может заключить договор на охрану нескольких квартир. Из-за ложных срабатываний сигнализации возможно несколько выездов патрульных экипажей по одной квартире. На владельца квартиры, вовремя не отключившего сигнализацию после своего прихода домой, налагается штраф, величина которого оговаривается при заключении договора охраны. Если отдел вневедомственной охраны не уберег имущество владельца квартиры, то он выплачивает пострадавшему заранее оговоренную сумму. От величины этой суммы зависит размер ежемесячной оплаты за охрану квартиры.

Необходимо создать базу данных, обеспечивающая деятельность отдела вневедомственной охраны квартир.

## 2.2 Группы пользователей информационной системы

В данной информационная системе можно выделить несколько пользователей:

* Случайный пользователь (Гость) – данный пользователь не может взаимодействовать с информационной системой.
* Конечный пользователь (потребитель информационной системы) – лицо или группа лиц, в интересах которых работает ИС. Данный пользователь был зарегистрирован в информационной системе и, как правило, не является программистом
* Коллектив специалистов (персонал информационной системы) – включающий администратора, командир экипажа и оператора.

## 2.3 Основные требования, предъявляемые к информационной системе

Данная информационная система должна соответствовать следующим требованиям:

* Гибкость – данная информационная система должна иметь возможность добавления новых таблиц.
* Надежность – данная информационная система должна выполнять свои функции.
* Эффективность – данная информационная система должна максимально быстро выполнять свои функции.
* Безопасность – данная информационная система должна быть защищен от взломов и иметь возможность восстановления и создания резерва.

Вывод по главе: была описана предметная область, также были описаны группы пользователей и основные требования, предъявляемые к информационной системе «Охрана квартир»,

# ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «Охрана квартир»

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

Перед началом разработки информационной системы необходимо создать визуальное представление о вариантах использования разрабатываемой системы.

Диаграмма вариантов использования является концептуальным представлением системы в процессе ее проектирования и разработки. Диаграмма вариантов использования описывает функциональное назначение системы или, другими словами, то, что система будет делать в процессе своего функционирования.

Цели построения диаграммы вариантов использования:

* определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования;
* сформулировать общие требования к функциональному проектированию системы;
* разработать исходную концептуальную модель системы для ее последующей реализации;
* подготовить документацию для взаимодействия разработчика системы с ее заказчиком и пользователями.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью, так называемых, вариантов использования. При этом актером или действующим лицом называется любая сущность, взаимодействующая с системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая может служить источником воздействия на моделируемую систему так, как определит сам разработчик.

В свою очередь, вариант использования служит для описания сервисов, которые система предоставляет актеру. Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемый системой при диалоге с актером.

В Приложении «А» на рисунке 4 представлена диаграмма вариантов использования для информационной системы «Охрана квартир». На этой диаграмме представлены следующие актеры:

1. Случайный пользователь (Гость) – может добавлять записи в таблицы: Client.
2. Конечный пользователь – может просматривать и читать таблицы: Capture, Contract, Flat, House, Prolonging, Calling. Также может добавлять записи в таблицы: Prolonging, Flat, House, Contract.
3. Командир экипажа – может просматривать и читать таблицы: House, Contract, Flat, Client, Flat-House, Settlement. И добавлять, изменять и удалять записи в таблице Capture, Calling.
4. Оператор – может изменять, удалять и добавлять данные во всех таблицах.
5. Администратор – может добавлять, удалять и изменять все таблицы в информационной системе. Также он может добавлять новых пользователей и разграничивать права доступа к базе данных. Это необходимо для того, чтобы сотрудники, не имеющие определенных полномочий, не могли получить доступ к различным объектам и данным.

### 3.1.2 Создание диаграммы деятельности

При моделировании поведения системы возникает необходимость не только представить процесс изменения ее состояний, но и детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Для этого применяется диаграмма деятельности.

Диаграмма деятельности — диаграмма, описывающая динамические аспекты системы. Диаграмма деятельности UML позволяет более детально визуализировать конкретный случай использования. Это поведенческая диаграмма, которая иллюстрирует поток деятельности через систему.

В Приложении «А» на рисунках с 5 - 9 находится диаграмма деятельности «Гость», которая показывает последовательность действий гостя, необходимых для достижения той или иной цели.

В Приложении «А» на рисунке 6 находится диаграмма деятельности «Гость», которая показывает последовательность действий гостя, необходимых для достижения той или иной цели.

В Приложении «А» на рисунке 7 находится диаграмма деятельности «Гость», которая показывает последовательность действий гостя, необходимых для достижения той или иной цели.

В Приложении «А» на рисунке 8 находится диаграмма деятельности «Гость», которая показывает последовательность действий гостя, необходимых для достижения той или иной цели.

В Приложении «А» на рисунке 9 находится диаграмма деятельности «Гость», которая показывает последовательность действий гостя, необходимых для достижения той или иной цели.

## 3.2 Разработка функциональной структуры информационной системы

Принципы модульного программирования во многом схожи с нисходящим проектированием, сначала определяется состав и подчиненность функции, а затем набор программных модулей, реализующих эти функции. Функции верхнего уровня обеспечиваются главным модулем, он управляет выполнением нижестоящих функций, которым соответствуют подчиненные модули. При определении набора модулей необходимо учитывать:

* каждый модуль вызывается на выполнение вышестоящим;
* принятие основных решений в алгоритме выносится на максимально высокий уровень по иерархии уровней;
* для использования одной и той же функции создается один модуль, который вызывается на выполнение по мере необходимости.

Карта навигации по информационной системе представлена в приложении «А» на рисунке 10 представлена карта переходов по информационной системе «Охрана квартир».

## 3.3 Разработка макета информационной системы

Прежде чем приступить к разработке информационной системы, сначала необходимо сориентироваться в элементах будущего интерфейса. Для этого уже определена задача, исследована предметная область, аудитория и то, как она будет пользоваться программным продуктом.

Пользовательский интерфейс — это внешний вид продукта, способ общения между пользователем и программой. Задача разработки интерфейса: сделать его полезным, интуитивно понятным и максимально простым.

Качественный дизайн повышает внешнюю привлекательность информационной системы за счет стратегической реализации таких элементов, как шрифты, цвета и изображения.

Так как разрабатываемая информационная система предназначена для широкого круга пользователей и предполагает разграничение прав доступа, необходима авторизация и регистрация.

На рисунке 11 представлен макет окна авторизации.

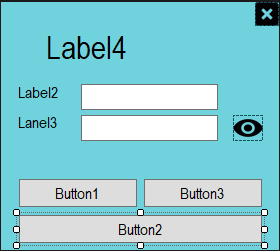


Рисунок 11 – Макет окна Авторизации

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

В случае если пользователь еще не зарегистрирован в системе, он может сделать это, нажав на кнопку «Зарегистрироваться». В этом случае откроется окно регистрации, представленное на рисунке 12.

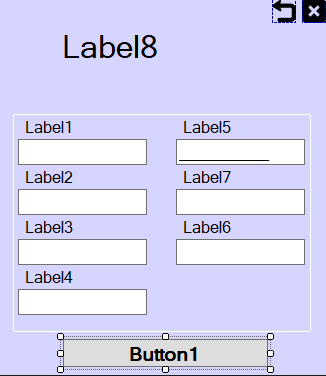


Рисунок 12 – Макет окна регистрации

На этапе регистрации пользователю необходимо придумать свой логин и пароль, а также ввести персональные данные, такие как фамилия, имя, отчество, номер телефона и адрес.

После того, как пользователь успешно прошел авторизацию, он может продолжить работу с таблицами на главной форме. На рисунке 13 - 16 представлена главная форма, которая будет отображена, в зависимости от прав пользователя в системе.



Рисунок 13 – Окно Администратора.



Рисунок 14 – Окно оператора

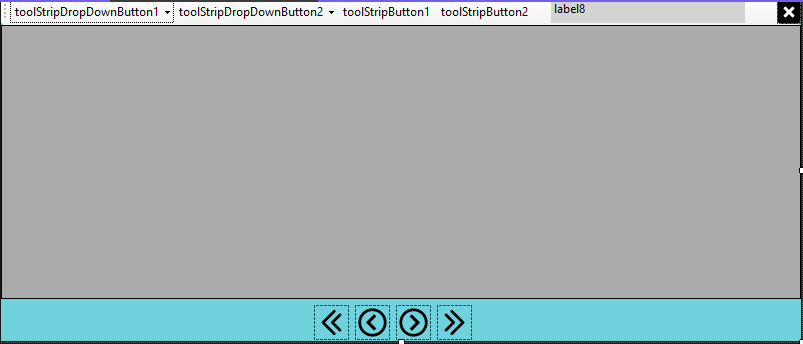


Рисунок 15 – Окно командира



Рисунок 16 – Окно Конечного пользователя

У конечного пользователя есть возможность зарегистрировать свою квартиру и это делается на отдельной форме. На рисунке 16 представлено окно регистрации квартиры.

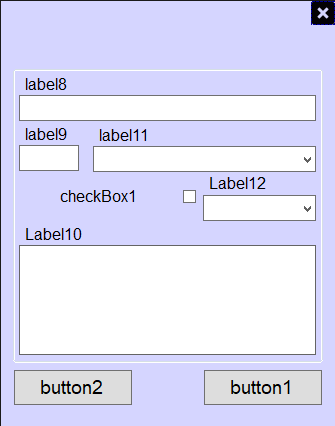


Рисунок 16 – Окно регистрации квартиры

Также конечный пользователь может поменять свои персональные данные, котоорые вводил на форме регистрации. На рисунке 17 представлена форма Профиля.

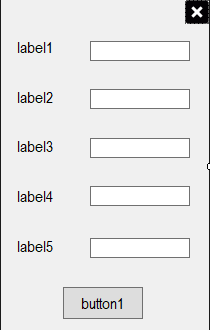


Рисунок 17 – форма Профиля клиента

Аналогичным образом в зависимости от роли пользователя выглядит главная форма, на которой расположены все необходимые таблицы. Например, командир может изменять только определённые таблицы.

Дизайн выполнен в минималистичном 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, карта переходов и макеты будущей информационной системы.

# РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ «Охрана квартир»

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 и набора, данных предоставленные вариантом учебной практики из Приложения «А» (рис. 1), можно выделить 11 типа сущностей (таблиц): Client, Flat, House, Contract, Tax, Compensation, Prolonging, Capture, Settlement, Flat-House, Calling.

**Описание сущностей**:

Один клиент, может иметь несколько квартир, поэтому между объектами Client и Settlement имеется связь «один-ко-многим». В одном доме может быть несколько квартир, поэтому между объектами Flat-House и House имеется связь «один-ко-многим». На одну квартиру можно заключить несколько договоров, поэтому между объектами Settlement лицо и Contract имеется связь «один-ко-многим». На одну квартиру может выехать несколько бригад на захват, поэтому между объектами Capture и Contract имеется связь «один-ко-многим». Договор можно несколько раз продлить, поэтому между объектами Contract и Prolonging имеется связь «один-ко-многим».

* к объекту Client относятся такие характеристики как: SecondName, FirstName, ThirdName, Phone, Address и Registr, который является полем первичного ключа;
* к объекту Flat относится: AddressFlat, Floor, TypeDoor, TypeBalcony, Balcony, Plan и FlatID, который является полем первичного ключа;
* к объекту House относится: Floors, Key, TypeHouse и HouseID, который является первичным ключом;
* к объекту Settlement относится: Registr, который является полем внешнего ключа и соединен с полем Registr в таблице Client, FlatID, который является полем внешнего ключа и соединен с полем FlatID в таблице Flat, SettlementID, который является полем первичного ключа;
* к объекту Flat-House относится: FlatID, который является полем внешнего ключа и соединен с полем FlatID в таблице Flat, HouseID, который является полем внешнего ключа и соединен с полем HouseID в таблице House, Flat-HouseID, который является полем первичного ключа;
* к объекту Contract относится: ContractName, DataStart, StopDate, Cost, SettlementID, который является полем внешнего ключа и соединен с полем SettlementID в таблице Settlement, TreatyID, который является полем первичного ключа;
* к объекту Prolonging относится: Prolong,Comment, TreatyID, который является полем внешнего ключа и соединен с полем TreatyID в таблице Contract, ProlongingID, который является полем первичного ключа;
* к объекту Capture относится: PatrollID, Chief, Brand, Document, ActionID, который является полем первичного ключа;
* к объекту Calling относится: DateTime, False, TreatyID, который является полем внешнего ключа и соединен с полем TreatyID в таблице Contract, ActionID, который является полем внешнего ключа и соединен с полем ActionID в таблице Capture, CallingID, который является полем первичного ключа;

В приложении «А» на рисунке 2 представлена логическая модель данных, созданная по этому набору данных.

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

Названия полей и их типы данных представлены в таблицах 2-11.

Таблица 1 – Поля таблицы «Client»

| Название поля | Тип данных |
| --- | --- |
| Registr | Числовой |
| SecondName | Текстовый |
| FirstName | Текстовый |
| ThirdName | Текстовый |
| Address | Текстовый |
| Phone | Текстовый |

Таблица 2 – Поля таблицы «Flat»

| Название поля | Тип данных |
| --- | --- |
| FlatID | Числовой |
| AddressFlat | Текстовый |
| Floor | Текстовый |
| TypeDoor | Текстовый |
| Balcony | Логический |
| TypeBalcony | Текстовый |
| Plane | Image |

Таблица 3 – Поля таблицы «House»

| Название поля | Тип данных |
| --- | --- |
| HouseID | Числовой |
| Floors | Текстовый |
| Key | Логический |
| TypeHouse | Текстовый |

Таблица 4 – Поля таблицы «Flat-House»

| Название поля | Тип данных |
| --- | --- |
| Flat-HouseID | Числовой |
| HouseID | Числовой |
| FlatID | Числовой |

Таблица 5 – Поля таблицы «Settlement»

| Название поля | Тип данных |
| --- | --- |
| SettlementID | Числовой |
| Registr | Числовой |
| FlatID | Числовой |

Таблица 6 – Поля таблицы «Contract»

| Название поля | Тип данных |
| --- | --- |
| TreatyID | Числовой |
| SettlementID | Числовой |
| ContractName | Текстовый |
| DateStart | Дата |
| DateStop | Дата |
| Cost | Денежный |

Таблица 7 – Поля таблицы «Prolonging»

| Название поля | Тип данных |
| --- | --- |
| ProlongingID | Числовой |
| TreatyID | Числовой |
| Prolong | Дата |
| Comment | Текстовый |

Таблица 8 – Поля таблицы «Capture»

| Название поля | Тип данных |
| --- | --- |
| ActionID | Числовой |
| PatrolID | Числовой |
| Chief | Текстовый |
| Brand | Текстовый |
| Document | Текстовый |

Таблица 9 – Поля таблицы «Calling»

| Название поля | Тип данных |
| --- | --- |
| CallingID | Числовой |
| TreatyID | Числовой |
| DateTime | Дата |
| False | Логический |
| TaxID | Числовой |
| CompensationID | Числовой |

Таблица 10 – Поля таблицы «User»

| Название поля | Тип данных |
| --- | --- |
| Registr | Числовой |
| Логин | Текстовый |
| Пароль | Текстовый |
| Права доступа | Текстовый |

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

В приложении «А» На рисунке 3 представлена физическая модель данных, отражающая все таблицы и их поля, а также связи между таблицами.

## 4.2. Разработка программных модулей информационной системы «Охрана квартир»

Модульное программирование — это организация программы как совокупности небольших блоков, называемых модулями, структура и поведение которых подчиняются определённым правилам. Использование модульного программирования позволяет упростить тестирование программы и обнаружение ошибок. Аппаратно-зависимые подзадачи могут быть строго отделены от других подзадач, что улучшает мобильность создаваемых программ.

Принцип модульности является средством упрощения задачи проектирования программного обеспечения и распределения процесса разработки. При разбиении ПО на модули для каждого модуля указывается реализуемая им функциональность, а также связи с другими модулями. Удобство использования модульной архитектуры заключается в возможности обновления или замены модуля, без необходимости изменения остальной системы.

Разработанные в MS Visual Studio формы продемонстрированы на рисунках 18 – 27.

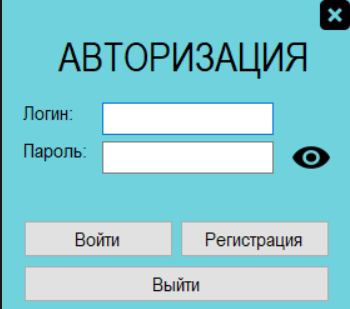


Рисунок 18 – Форма «Авторизации»

Свойства элементов формы «Авторизация» представлены в таблице 11.

Таблица 11 – Свойства элементов формы «Авторизация».

| Объект | Свойство | Значение |
| --- | --- | --- |
| Авторизация | FormBorderStyle | None |
| BackColor | 111; 210; 221 |
| Text | «Авторизация» |
| Size | 371; 305 |
| StartPosition | CenterScreen |
| Label1 | Text | «» |
| Font | Arial Narrow; 12pt |
| Label2 | Text | «Логин:» |
| Font | Arial Narrow; 12pt |
| Label3 | Text | «Пароль:» |
| Font | Arial Narrow; 12pt |
| Label4 | Text | «АВТОРИЗАЦИЯ» |
| Font | Arial Narrow; 12pt |
| TextBox1 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Arial Narrow; 12pt |
| TextBox2 | PasswordChar | «\*» |
| ShortcutsEnabled | false |
| Font | Arial Narrow; 12pt |
| PictureBox1 | BackColor | 111; 210; 221 |
| PictureBox2 | BackColor | 111; 210; 221 |
| Button1 | Text | «Войти» |
| Font | Arial Narrow; 12pt |
| Button2 | Text | «Выйти» |
| Font | Arial Narrow; 12pt |
| Button3 | Text | «Регистрация» |
| Font | Arial Narrow; 12pt |

Событийно-управляемые процедуры в форме авторизации представлены в таблице 12

Таблица 12 – Событийно-управляемые процедуры в форме «Авторизация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Авторизация | public partial class Авторизация : Form  {  public int ogr = 5;  int click = 0;  Point Mouse;  public Авторизация()  {  InitializeComponent();  } |
| button1\_Click | private void button1\_Click(object sender, EventArgs e)  {  if (textBox1.Text == "" && textBox2.Text == "")  {  MessageBox.Show("Введите и Логин и Пароль", "Ошибка");  }  else if (ogr == 0)  {  MessageBox.Show("Количество ваших попыток было исчерпанно\nОбратитись к администратору и закройте форму авторизации");  button1.Visible = false;  button3.Visible = false;  label2.Visible = false;  label1.Visible = false;  label3.Visible = false;  textBox1.Visible = false;  textBox2.Visible = false;  pictureBox1.Visible = false;  }  else  {  using (SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog =уП\_ПМ\_01\_Неверов\_ДС; Integrated Security = True"))  {  SqlDataAdapter da = new SqlDataAdapter("select \* from [User]", sqlConnect);  DataTable dt = new DataTable();  da.Fill(dt);  Boolean логин = false;  int g = 0;  Boolean пароль = false;  for (int i = 0; i < dt.Rows.Count; i++)  {  if ((dt.Rows[i]["Логин"].ToString() == textBox1.Text) && (dt.Rows[i]["Пароль"].ToString() == textBox2.Text))  {  логин = true; пароль = true; g = i;  }  if ((dt.Rows[i]["Логин"].ToString() != textBox1.Text) && (dt.Rows[i]["Пароль"].ToString() == textBox2.Text))  {  логин = false; пароль = true;  }  if ((dt.Rows[i]["Логин"].ToString() == textBox1.Text) && (dt.Rows[i]["Пароль"].ToString() != textBox2.Text))  {  логин = true; пароль = false;  }  }  if (логин == true && пароль == true)  {  switch (dt.Rows[g]["Права доступа"].ToString())  {  case "КП":  this.Hide();  КП f4 = new КП();  f4.Show();  f4.registr(dt.Rows[g]["Registr"].ToString());  break;  case "Администратор":  this.Hide();  Администратор f57 = new Администратор();  f57.Show();  f57.a("Админ");  break;  case "Командир":  Командир f2 = new Командир();  this.Hide();  f2.Show();  break;  case "Оператор":  this.Hide();  Администратор f5 = new Администратор();  f5.Show();  f5.a("Опер");  break;  }  }  else if (логин == false || пароль == false)  {  MessageBox.Show("Неправельные Логин или Пароль, попробуйте введите еще раз");  ogr = ogr - 1;  label1.Text = $"Для входа осталось попыток: {ogr}";  }  }  }  } |
| pictureBox1\_Click | private void pictureBox1\_Click(object sender, EventArgs e)  {  click += 1;  if (click==1)  {  pictureBox1.ImageLocation = $"//apetfs/ИСПП-5/Неверов\_ДС/УП ПМ 01/Охрана квартир/Охрана квартир/5618419.png";  textBox2.PasswordChar = '\0';  }  else if (click == 2)  {  pictureBox1.ImageLocation = $"//apetfs/ИСПП-5/Неверов\_ДС/УП ПМ 01/Охрана квартир/Охрана квартир/free-icon-visibility-button-60809.png";  textBox2.PasswordChar = '\*';  click = 0;  }  } |
| textBox1\_KeyPress | private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  textBox2.Focus();  }  } |
| textBox2\_KeyPress | private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  button1.Focus();  }  } |
| button2\_Click | private void button2\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| button3\_Click | private void button3\_Click(object sender, EventArgs e)  {  Регистрация а1 = new Регистрация();  Hide();  а1.Show();  } |
| Form1\_Load | private void Form1\_Load(object sender, EventArgs e)  {  ToolTip tt = new ToolTip();  tt.SetToolTip(button3, "Нажмите, чтобы оформить документ и зарегестрироваться");  tt.SetToolTip(pictureBox1, "Нажмите, чтобы просмотреть пароль");  } |
| pictureBox2\_Click | private void pictureBox2\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| Авторизация\_MouseDown | private void Авторизация\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |
| Авторизация\_MouseMove | private void Авторизация\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |

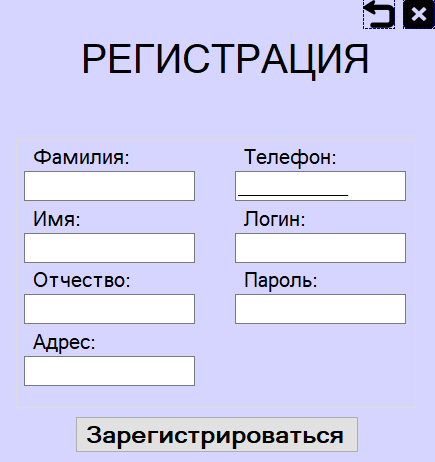


Рисунок 19 – Форма «Регистрация»

Свойства элементов формы «Авторизация» представлены в таблице 13.

Таблица 13 – Свойства элементов формы «Авторизация».

| Объект | Свойство | Значение |
| --- | --- | --- |
| Регистрация | FormBorderStyle | None |
| BackColor | 213; 213; 255 |
| Text | «Регистрация» |
| Size | 435; 462 |
| StartPosition | CenterScreen |
| Label1 | Text | «Фамилия:» |
| Font | Microsoft Sans Serif; 12pt |
| Label2 | Text | «Имя:» |
| Font | Microsoft Sans Serif; 12pt |
| Label3 | Text | «Отчество:» |
| Font | Microsoft Sans Serif; 12pt |
| Label4 | Text | «Адрес:» |
| Font | Microsoft Sans Serif; 12pt |
| Label5 | Text | «Телефон:» |
| Font | Microsoft Sans Serif; 12pt |
| Label6 | Text | «Логин:» |
| Font | Microsoft Sans Serif; 12pt |
| Label7 | Text | «Пароль:» |
| Font | Microsoft Sans Serif; 12pt |
| Label8 | Text | «РЕГИСТРАЦИЯ» |
| Font | Microsoft Sans Serif; 24pt |
| Label9 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| GroupBox1 | Text | «» |
| Size | 400; 277 |
| TextBox1 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox2 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox3 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox4 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox5 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox6 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaskedTextBox1 | Mask | 9990000000 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| PictureBox1 | BackColor | 213; 213; 255 |
| PictureBox2 | BackColor | 213; 213; 255 |
| Button1 | Text | «Зарегистрироваться» |
| Font | Arial Narrow; 12pt |

Событийно-управляемые процедуры в форме авторизации представлены в таблице 14

Таблица 14 – Событийно-управляемые процедуры в форме «Регистрация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Регистрация | public partial class Регистрация : Form  {  Point Mouse;  public Регистрация()  {  InitializeComponent();  } |
| Регистрация\_Load | private void Регистрация\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП\_ПМ\_01\_Неверов\_ДСDataSet1.Client". При необходимости она может быть перемещена или удалена.  this.clientTableAdapter1.Fill(this.уП\_ПМ\_01\_Неверов\_ДСDataSet1.Client);  label9.Text = "Введите свои персональные данные";  this.Size = new Size(325, 375);  ToolTip tt = new ToolTip();  tt.SetToolTip(pictureBox1, "Нажмите, чтобы закрыть приложение");  tt.SetToolTip(pictureBox2, "Нажмите, чтобы вернуться к форме авторизации");  } |
| button1\_Click | private void button1\_Click(object sender, EventArgs e)  {  using (SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = УП\_ПМ\_01\_Неверов\_ДС; Integrated Security = True"))  {  SqlDataAdapter da = new SqlDataAdapter("select \* from [User]", sqlConnect);  DataTable dt = new DataTable();  da.Fill(dt);  Boolean логин = true;  int n = dataGridView1.RowCount;  for (int i = 0; i < dt.Rows.Count; i++)  {  if (dt.Rows[i]["Логин"].ToString() == textBox5.Text)  {  логин = false;  }  }  if (textBox1.Text == "" || textBox2.Text == "" || textBox4.Text == "" || maskedTextBox1.Text == "" || textBox5.Text == "" || textBox6.Text == "")  {  MessageBox.Show("Введите все данные!");  }  else  {  if (логин == false)  {  MessageBox.Show("Пользователь уже существует!!!");  }  else  {  SqlDataAdapter info1 = new SqlDataAdapter($"INSERT INTO [Client] (Registr,SecondName,FirstName,ThirdName,Address,Phone)VALUES('{n}','{textBox1.Text}','{textBox2.Text}','{textBox3.Text}','{textBox4.Text}','{maskedTextBox1.Text}');", sqlConnect);  SqlDataAdapter info2 = new SqlDataAdapter($"INSERT INTO [User] (Registr,Логин,Пароль)VALUES('{n}','{textBox5.Text}','{textBox6.Text}');", sqlConnect);  DataTable dt1 = new DataTable();  DataTable dt2 = new DataTable();  info1.Fill(dt1);  info2.Fill(dt2);  }  MessageBox.Show($"Пользователь {textBox2.Text} {textBox1.Text} {textBox3.Text} был зарегестрирован.\nСейчас вас перекинет на форму авторизации, где вам необходимо ввести логин и пароль, который вы вводили.");  Авторизация fd = new Авторизация();  Hide();  fd.Show();  }  }  } |
| pictureBox1\_Click | private void pictureBox1\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| pictureBox2\_Click | private void pictureBox2\_Click(object sender, EventArgs e)  {  Авторизация fd = new Авторизация();  Hide();  fd.Show();  } |
| textBox1\_KeyPress | private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  textBox2.Focus();  }  } |
| textBox2\_KeyPress | private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  textBox3.Focus();  }  } |
| textBox3\_KeyPress | private void textBox3\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  textBox4.Focus();  }  } |
| textBox4\_KeyPress | private void textBox4\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  maskedTextBox1.Focus();  }  } |
| maskedTextBox1\_KeyPress | private void maskedTextBox1\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  textBox5.Focus();  }  } |
| textBox7\_KeyPress | private void textBox7\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  textBox6.Focus();  }  } |
| textBox6\_KeyPress | private void textBox6\_KeyPress(object sender, KeyPressEventArgs e)  {  if (e.KeyChar == (char)Keys.Enter)  {  button1.Focus();  }  } |
| Регистрация\_MouseMove | private void Регистрация\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Регистрация\_MouseDown | private void Регистрация\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |

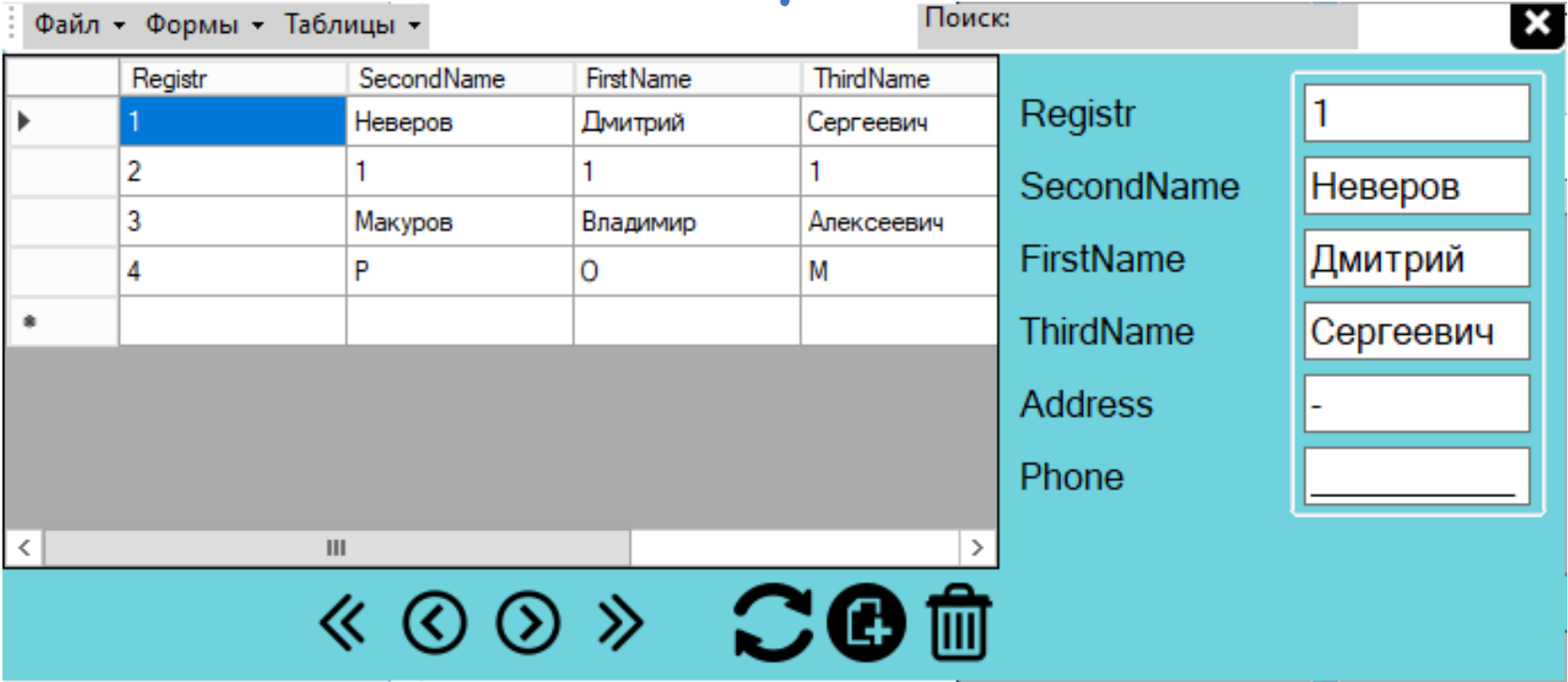


Рисунок 20 – Форма «Администратора»

Свойства элементов формы «Авторизация» представлены в таблице 15.

Таблица 15 – Свойства элементов формы «Администратор».

| Объект | Свойство | Значение |
| --- | --- | --- |
| Администратор | FormBorderStyle | None |
| BackColor | 111; 210; 221 |
| Text | «Форма Администратор» |
| Size | 688, 301 |
| StartPosition | CenterScreen |
| ToolStrip1 | Items | toolStripDropDownButton1 |
| toolStripDropDownButton2 |
| toolStripDropDownButton3 |
| OpenFileDialog1 | Filter | «Image files (\*.BMP, \*.JPG, \*.GIF,\*.PNG)|\*.bmp;\*.jpg;\*.gif;\*.png» |
| DataGridView1 | Size | 585; 279 |
| ComboBox1 | Items | «Отдельный», «Совмещенный» |
| Font | Microsoft Sans Serif; 12pt |
| ComboBox2 | Items | «Металическая», «Деревянная», «Две двери» |
| Font | Microsoft Sans Serif; 12pt |
| ComboBox3 | Items | «Кирпичный», «Панельный» |
| Font | Microsoft Sans Serif; 12pt |
| ComboBox4 | Items | «Администратор», «Оператор», «Командир», «КП» |
| Font | Microsoft Sans Serif; 12pt |
| CheckBox1 | Text | «Есть балкон» |
| Font | Arial Narrow; 12pt |
| RightToLeft | Yes |
| CheckBox2 | Text | «Есть замок» |
| Font | Arial Narrow; 12pt |
| RightToLeft | Yes |
| CheckBox3 | Text | «?Ложный» |
| Font | Arial Narrow; 12pt |
| RightToLeft | Yes |
| GroupBox1 | Text | «» |
| Size | 151; 250 |
| GroupBox2 | Text | «» |
| Size | 151; 250 |
| GroupBox3 | Text | «» |
| Size | 151; 250 |
| GroupBox4 | Text | «» |
| Size | 151; 250 |
| GroupBox5 | Text | «» |
| Size | 151; 250 |
| GroupBox6 | Text | «» |
| Size | 151; 250 |
| GroupBox7 | Text | «» |
| Size | 151; 250 |
| GroupBox8 | Text | «» |
| Size | 151; 250 |
| GroupBox9 | Text | «» |
| Size | 164; 289 |
| GroupBox10 | Text | «» |
| Size | 151; 250 |
| Label1 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label2 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label3 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label4 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label5 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label6 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label7 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label8 | Text | «Поиск:» |
| Font | Segoe UI; 9pt |
| Label9 | Text | «» |
| Font | Microsoft Sans Serif; 7,8pt |
| Label10 | Text | «» |
| Font | Microsoft Sans Serif; 7,8pt |
| Label11 | Text | «» |
| Font | Microsoft Sans Serif; 12pt |
| Label12 | Text | «» |
| Font | Microsoft Sans Serif; 7,8pt |
| Label13 | Text | «» |
| Font | Microsoft Sans Serif; 7,8pt |
| Label14 | Text | «» |
| Font | Microsoft Sans Serif; 7,8pt |
| Label15 | Text | «» |
| Font | Microsoft Sans Serif; 7,8pt |
| TextBox1 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox2 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox3 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox4 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox5 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox6 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox7 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox8 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox9 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox10 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox11 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox12 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox13 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox14 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox15 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox16 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox17 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox18 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox19 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox20 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox21 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox22 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox23 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox24 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox25 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox26 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox27 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox28 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox29 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox30 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox31 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox32 | ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaxLength | 60 |
| TextBox33 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox34 | MaxLength | 60 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox35 | MaxLength | 50 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| TextBox36 | MaxLength | 20 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaskedTextBox1 | Mask | 9990000000 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaskedTextBox2 | Mask | 00/00/0000 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaskedTextBox3 | Mask | 00/00/0000 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaskedTextBox4 | Mask | 00/00/0000 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| MaskedTextBox5 | Mask | 00/00/0000 |
| ShortcutsEnabled | false |
| Font | Microsoft Sans Serif; 12pt |
| PictureBox1 | BackColor | White |
| SizeMode | StretchImage |
| PictureBox2 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox3 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox4 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox5 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox6 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox7 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox8 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |
| PictureBox9 | BackColor | 111; 210; 221 |
| SizeMode | StretchImage |

Событийно-управляемые процедуры в форме авторизации представлены в таблице 16.

Таблица 16 – Событийно-управляемые процедуры в форме «Регистрация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Регистрация | public partial class Регистрация : Form  {  Point Mouse;  public Регистрация()  {  InitializeComponent();  } |

Вывод по главе: была спроектирована и разработана система базы данных, были полностью спроектированы и разработаны все программные модули информационной системы «Охрана квартир».

# 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. 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

## 5.3 Результаты проведения тестирования

# ЗАКЛЮЧЕНИЕ

# 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.).

# ПРИЛОЖЕНИЕ A



Рисунок 1 - Набор данных к информационной системе «Охрана квартир»

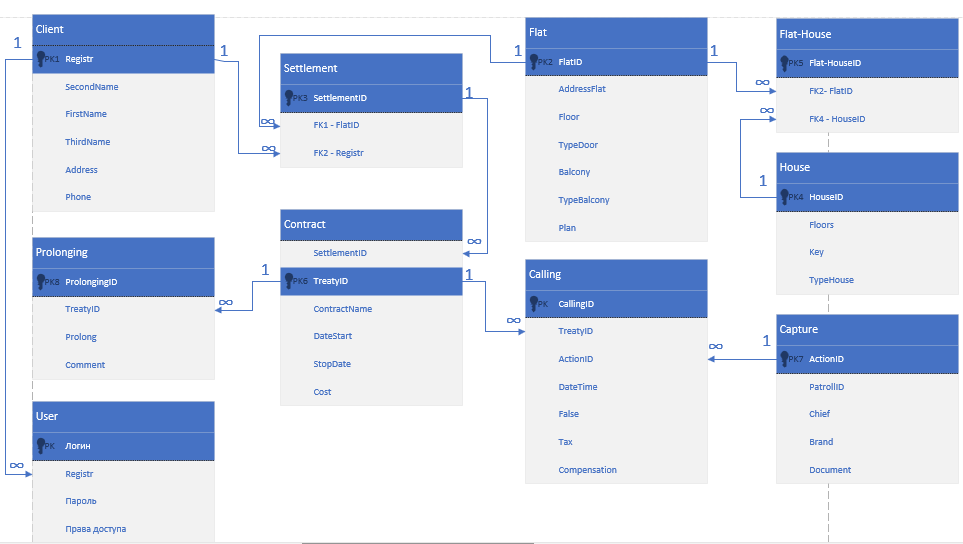


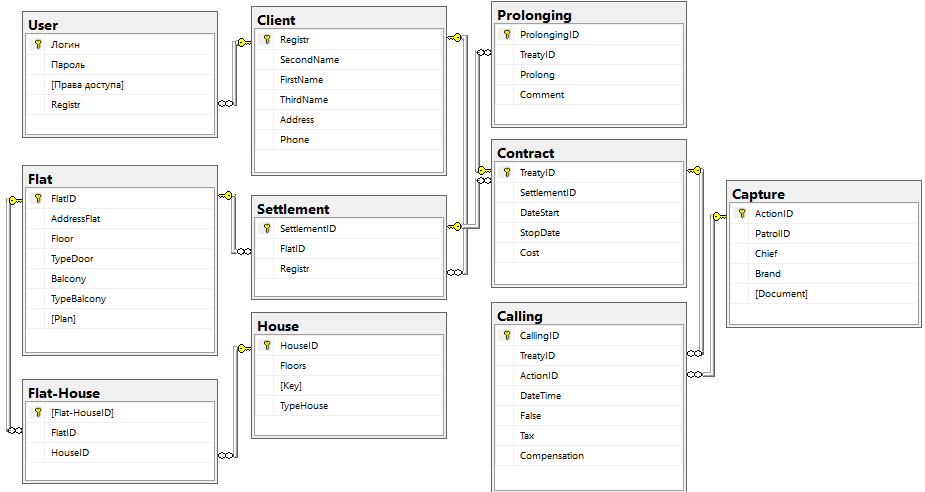
Рисунок 2 – Логическая модель данных 

Рисунок 3 – Физическая модель данных

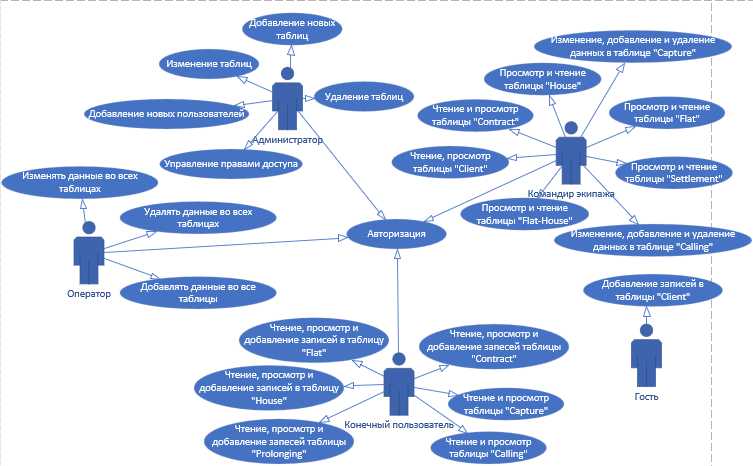


Рисунок 4 – Диаграмма вариантов использования



Рисунок 5 – Диаграмма деятельности Администратора



Рисунок 6 – Диаграмма деятельности Гостя



Рисунок 7 – Диаграмма деятельности Гостя



Рисунок 8 – Диаграмма деятельности Гостя



Рисунок 9 – Диаграмма деятельности Гостя

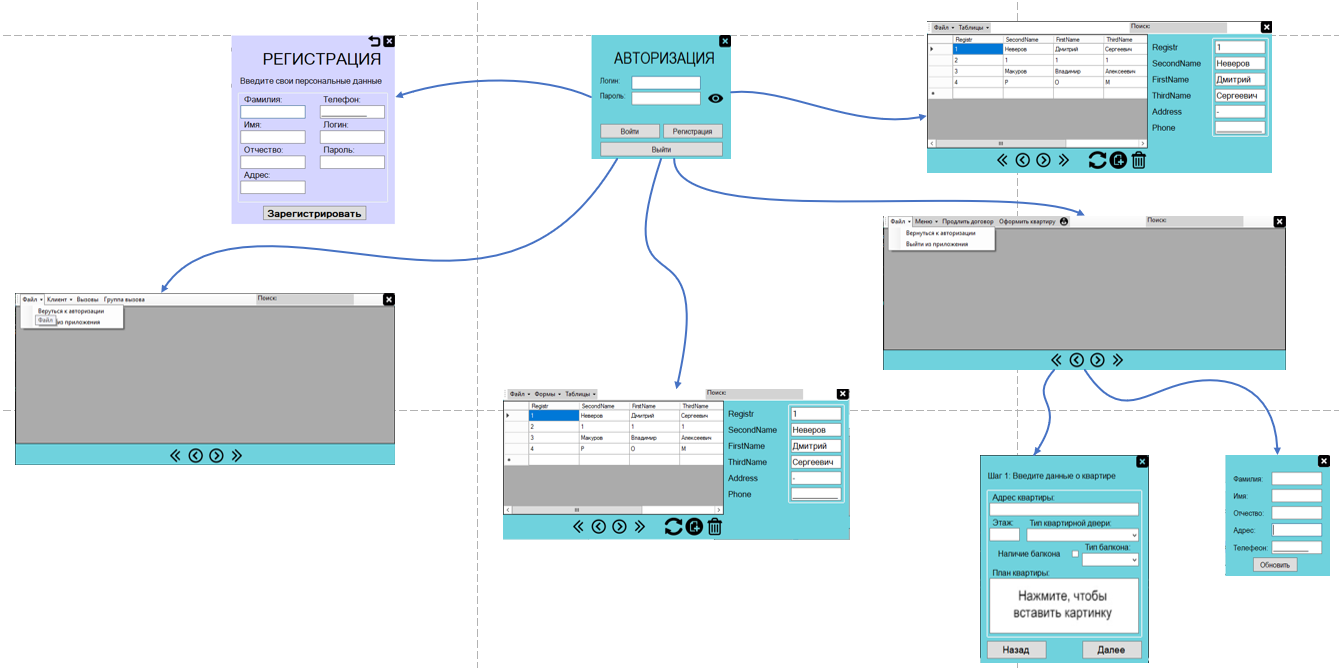


Рисунок 10 – Карта переходов

# ПРИЛОЖЕНИЕ Б