**Содержание**

[Введение 4](#_Toc160175774)

[1. Предварительное планирование и проверка осуществимости 6](#_Toc160175775)

[1.1. Требования к системе в целом 6](#_Toc160175776)

[1.2. Требования к функциям, выполняемым системой 8](#_Toc160175777)

[1.3. Требования к видам обеспечения 9](#_Toc160175778)

[1.4. Сравнение и анализ аналогов системы 10](#_Toc160175779)

[1.5. Инфологическая модель предметной области 13](#_Toc160175780)

[1.6. Прототипирование интерфейса 14](#_Toc160175781)

[2. Практическая часть 15](#_Toc160175782)

[2.1. Назначение и цели создания системы 15](#_Toc160175783)

[2.2. Характеристики объекта автоматизации 15](#_Toc160175784)

[2.3. Обзор инструментальных средств создания продукта 16](#_Toc160175785)

[2.4. Обоснование выбора средств реализации ИС 18](#_Toc160175786)

[2.5 Описание логики функционирования ПО 20](#_Toc160175787)

[2.6 Реализация интерфейса ПО 25](#_Toc160175788)

[2.7. Написание программного кода 30](#_Toc160175789)

[2.8 Защита информации 32](#_Toc160175790)

[Заключение 33](#_Toc160175791)

[Список использованных источников 34](#_Toc160175792)

# Введение

"Разработка модуля информационной системы для ГИБДД" - это актуальная тема в современном мире, где эффективное управление дорожным движением и обеспечение безопасности на дорогах играют ключевую роль. Государственная инспекция безопасности дорожного движения (ГИБДД) является важным государственным органом, ответственным за контроль и регулирование дорожной деятельности в России. Разработка модуля информационной системы для ГИБДД имеет целью повысить эффективность работы этого органа, путем улучшения контроля и автоматизации процесса мониторинг дорожного движения, а также обеспечить оперативный доступ к информации о дорожной обстановке.

В данной курсовой работе рассматриваются проблемы и аспекты разработки такой информационной системы, её потенциальные преимущества для государственных органов и граждан, а также возможные технологические и организационные решения, способствующие реализации данной системы

**Актуальность темы** – в современном мире уровень дорожных происшествий и нарушений в области дорожного движения растет с каждым годом. Это требует от правоохранительных органов, включая ГИБДД, обновления и совершенствования системы контроля и управления дорожным движением.

**Цель** – создать модуль информационной системы, который будет иметь в себе объёмный функционал для сотрудников ГИБДД, который автоматизирует выписку и поиск штрафов.

**Объект** – государственная инспекция безопасности дорожного движения (ГИБДД) в России.

**Предмет** – модуль информационной системы для ГИБДД

**Для достижения цели, требуется выполнить следующие задачи:**

1. Провести предварительное планирование и проверку осуществимости.
2. Определить требования к программному продукту.
3. Разработать прототип приложения.
4. Реализовать приложение в выбранной среде IDE.
5. Разместить проект на веб-сервисе GitHub или аналогичном.

# Предварительное планирование и проверка осуществимости

# 1.1. Требования к системе в целом

Требования к структуре и функционированию системе

Система должна состоять из сервера баз данных, приложения.

Сервер БД отвечает за хранение базы данных. Приложение организует работу базы данных и обмен данными с сервером баз данных. Приложение – основная часть системы – обрабатывает данные полученные от сервера БД и выводит их в понятном для конечного пользователя виде, а также формирует запросы от пользователя к серверу БД.

Требования к численности и квалификации персонала

Пользователями приложения являются как потенциальные клиенты, которые хотят оформить свое авто, так и нарушители, которые получили штраф за нарушение правил дорожного движения. Пользователи используют систему через персональный компьютер, поэтому количество людей, желающих просмотреть приложение или воспользоваться услугами неограниченно.

Обслуживающий персонал:

• администратор – 1 человек;

• модератор – 1 человек.

• диспетчер – 1 человек.

Пользователи должны иметь минимальные навыки работы на компьютере.

Обслуживающий персонал должен иметь навыки работы на компьютере в среде Microsoft Windows (или другой операционной системе(ОС)).

Администратор сервера должен иметь опыт работы с SQL и Visual Studio 19 знать язык программирования C#, WPF.

Требования к надежности

Программное обеспечение системы должно функционировать и не приводить к зависанию системы при любых ошибочных действиях операторов в среде данного программного обеспечения.

Требования к эргономике и технической эстетике

Программное обеспечение и технические средства должны иметь эстетически привлекательный вид и быть удобными в использовании.

Требования к эксплуатации и техническому обслуживанию

Для нормальной работы системы необходимо произвести ряд действий

На SQL-сервере:

• Установить лицензионную копию ОС;

• Запустить программу установки SQL-сервера.

В приложении:

• Установить лицензионную копию ОС;

• Установить копию приложения в формате .exe;

• Запустить программу установки приложения.

Требования к допустимым площадям размещения ИС, к параметрам сети энергоснабжения и т.д. определяются требованиями использующегося аппаратного обеспечения.

Требования к защите информации от несанкционированного доступа

Защита ИС и информации, с которой она работает, от несанкционированного доступа возлагается частично на операционную систему и специализированные программные и программно–аппаратные продукты, частично на систему аутентификации, реализованную в компонентах ИС. Доступ к данным предоставляется только легальным пользователям. В процессе аутентификации имя пользователя и его пароль передаются в базу данных, обеспечивающему защиту передаваемых данных путем шифрования.

Требования по сохранности информации при авариях

Сохранность данных должна быть обеспеченна:

• При сбоях в работе оборудования сети в любой момент времени;

• При сбоях электропитания или программного обеспечения не ведущих к физическому уничтожению или повреждению информации.

Требования к защите от влияния внешних воздействий

Автоматизированная система должна быть устойчивой к наличию и параллельной с ней работе на той же ЭВМ другого программного обеспечения.

Для уменьшения влияния перебоев с электропитанием рекомендуется питание «Сервера БД» и «Приложение» производить через блок бесперебойного питания.

Требования к стандартизации и унификации

Автоматизированная система должна предоставлять пользователю привычный, общепринятый в среде Microsoft Windows интерфейс. Программная документация, поставляемая с программой, должна быть оформлена в соответствии со стандартом ЕСПД.

# 1.2. Требования к функциям, выполняемым системой

Функции подсистемы «Сервер базы данных»:

• Получение запросов от «пользователя БД», расположенного на SQL-сервере;

• Проверка легальности выполнения запроса на основе идентификационной информации, вложенной в запрос;

• Выполнение SQL-запросов к БД;

• Передача результата SQL-запроса «пользователю БД».

Функции подсистемы «Приложение»:

• Передача пользователю следующей информации:

• Информация о разделах;

• Информация о водителе.

• Изменение информации:

• Изменение информации о разделах;

• Формирование и передача запросов серверу БД;

• Получение от сервера БД результатов запросов.

Функции подсистемы «пользователи базы данных»:

• Получение запросов приложения;

• Получение от сервера БД результатов запросов.

# 1.3. Требования к видам обеспечения

Требования к математическому обеспечению

В основе разрабатываемого ПО должна лежать реляционная модель базы данных. Все таблицы должны находиться в третьей или четвертой нормальной форме. Связи между таблицами должны быть вида: «один к одному» или «один ко многим». При проектировании используется метод «сущность – связь».

Требования к информационному обеспечению

Данные в системе должны быть организованны в виде связанных таблиц. Хранение и доступ к данным осуществляется с применением реляционные систем управления базами данных. Информационный обмен между компонентами системы осуществляется по технологии пользователь-сервер. Желательно обеспечить возможность резервного копирования и восстановления данных.

Требования к лингвистическому обеспечению

В качестве языка программирования используются – C#, SQL взаимодействие пользователя и ПО осуществляется через оконный интерфейс.

Требования к программному обеспечению

Система должна иметь следующее программное обеспечение: ОС Windows 10 и выше и SQL Server.

Требования к техническому обеспечению

Для работоспособности сайта необходимы следующие программные и технические средства:

• Клавиатура;

• Манипулятор мышь;

• Процессор Intel Core i3 2200 или AMD Radeon 7 2700;

• Оперативная память 4GB и больше;

• Видеокарта 2 Гб и больше;

• Монитор, поддерживающего разрешение 800х600 точек и больше (рекомендуется 1024х768 точек) при глубине цвета 16 бит и более;

• Дисковое пространство 5 Гб и больше;

# 1.4. Сравнение и анализ аналогов системы

Приложение «Мои Штрафы: проверка и оплата»

* Разработчик: ООО "ИнфоБокс"

Особенности:

* + Проверка штрафов ГИБДД по номеру водительского удостоверения или СТС
  + Оплата штрафов с помощью банковской карты
  + Уведомления о новых штрафах
  + История оплаченных штрафов
  + Возможность обжаловать штрафы

Интерфейс приложения «Мои штрафы» представлен на рисунке 1.1



Рисунок 1.1 – Пример приложения «Мои Штрафы: проверка и оплата»

Штрафы ГИБДД официальные ПДД

* Разработчик: ГУОБДД МВД России
* Особенности:
  + Проверка штрафов ГИБДД по номеру постановления, номеру водительского удостоверения или СТС
  + Оплата штрафов с помощью банковской карты или электронного кошелька
  + Уведомления о новых штрафах
  + История оплаченных штрафов
  + Информация о правилах дорожного движения
  + Тесты на знание ПДД

Интерфейс приложения «Мои штрафы» представлен на рисунке на рисунке 1.2

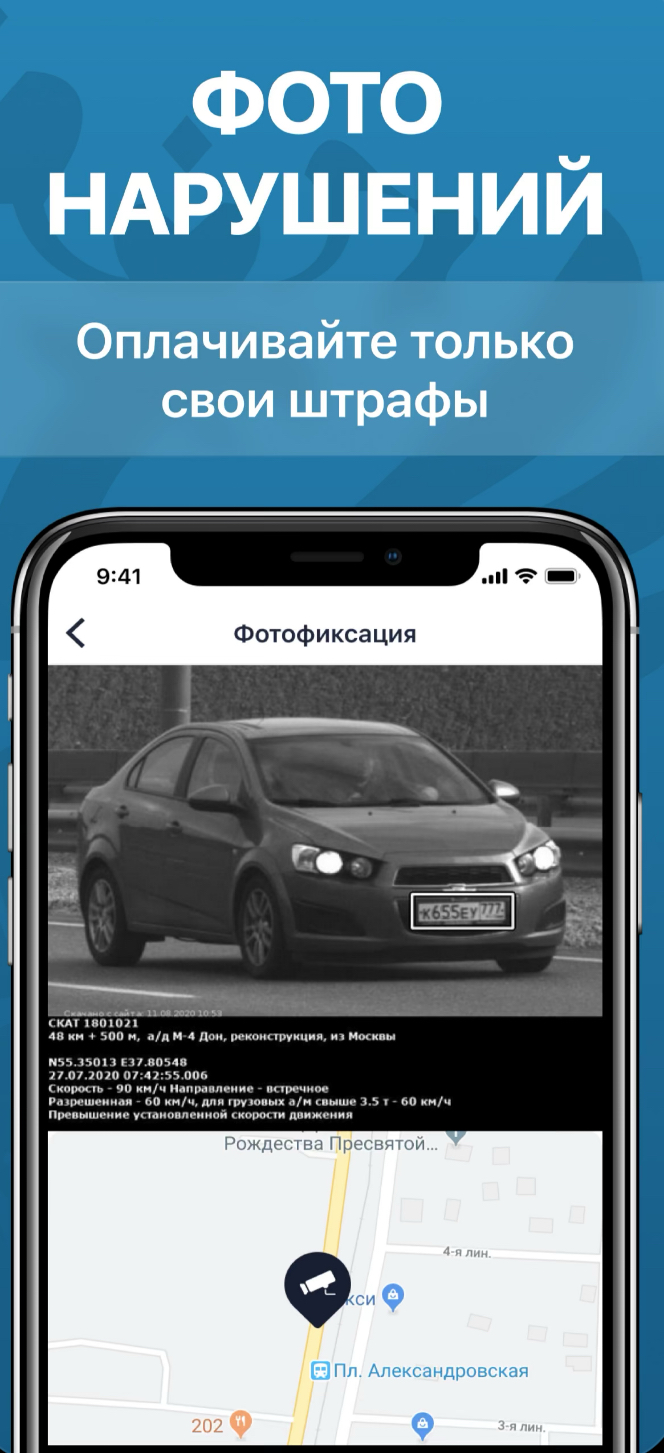


Рисунок 1.2 – Пример приложения «Штрафы ГИБДД официальные ПДД»

Таблица 1.1 Сравнение приложений

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристика** | **Мои Штрафы** | **Штрафы ГИБДД** | **Мое приложение** |
| Разработчик | ООО "ИнфоБокс" | ГУОБДД МВД России | Талызенков С.П. |
| Проверка штрафов | По номеру ВУ или СТС | По номеру постановления, ВУ или СТС | По номеру постановления, ВУ или СТС |
| Оплата штрафов | Банковская карта | Банковская карта, электронный кошелек | Банковская карта, электронный кошелек |
| Уведомления о новых штрафах | Да | Да | Да |
| История оплаченных штрафов | Да | Да | Да |
| Обжалование штрафов | Да | Нет | Нет |
| Информация о ПДД | Нет | Да | Нет |
| Тесты на знание ПДД | Нет | Да | Нет |

**Таким образом, оценивая конкурентные приложения и мое, могу сделать вывод, что** Оба приложения предоставляют возможность проверки и оплаты штрафов ГИБДД. Приложение "Мои Штрафы" предлагает дополнительную функцию обжалования штрафов, а приложение "Штрафы ГИБДД" предоставляет информацию о ПДД и позволяет проходить тесты на знание правил. Выбор приложения зависит от индивидуальных потребностей пользователя.

# 1.5. Инфологическая модель предметной области

Инфологическая модель предметной области — это концептуальная модель, которая описывает предметную область с точки зрения информационных аспектов. Она служит основой для разработки базы данных и представляет собой абстрактное представление данных, используемых и обрабатываемых в рамках данной предметной области.

Основными элементами инфологической модели являются сущности, атрибуты и отношения. Сущности представляют объекты или понятия, обладающие определенными характеристиками, атрибутами. Атрибуты описывают свойства сущностей и могут иметь различные типы данных. Отношения определяют связи и взаимодействия между сущностями.

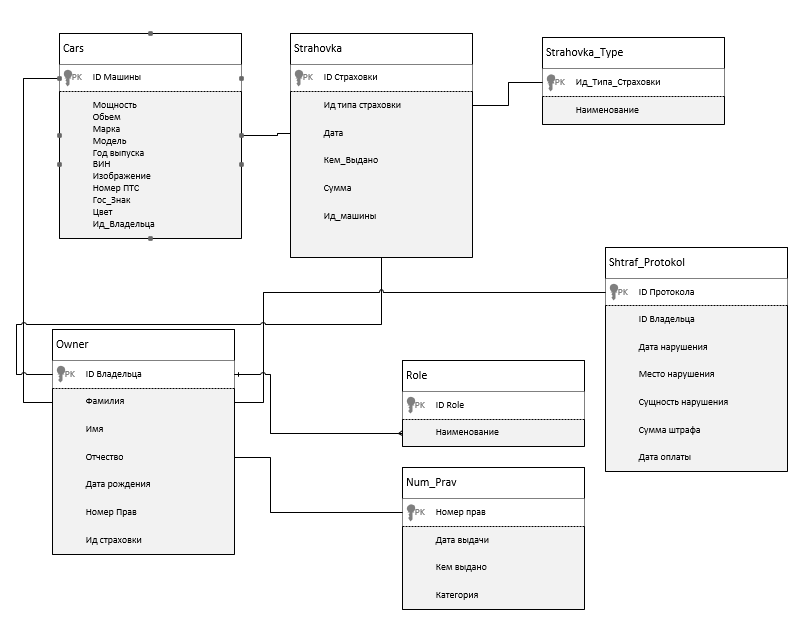


Рисунок 1.3 – База данных приложения.

# 1.6. Прототипирование интерфейса

В прототипе интерфейса изображенном на рисунке 1.4. навигация по приложению совершается за счет 5 вкладок которым будет присвоено свое наименование для понимания за что отвечает выбранный раздел.

Страница Главная

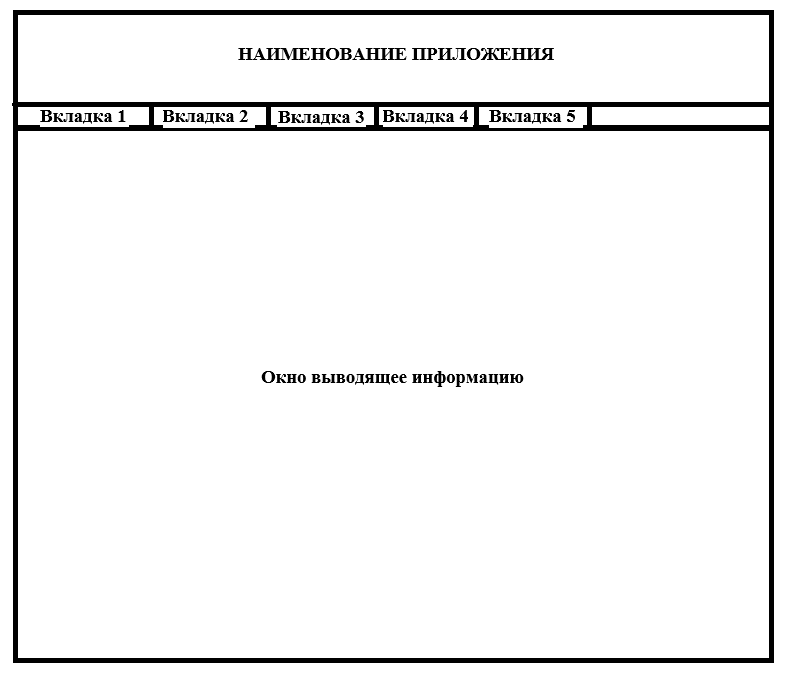


Рисунок 1.4 – Прототип страницы Главная

Страница авторизации, изображенная на рисунке 1.5. работает по легальным пользователям которые указали данные для регистрации к авторизации присоединена база данных которая хранит в себе логины и пароли пользователей тем самым обеспечивая конфиденциальность информации.

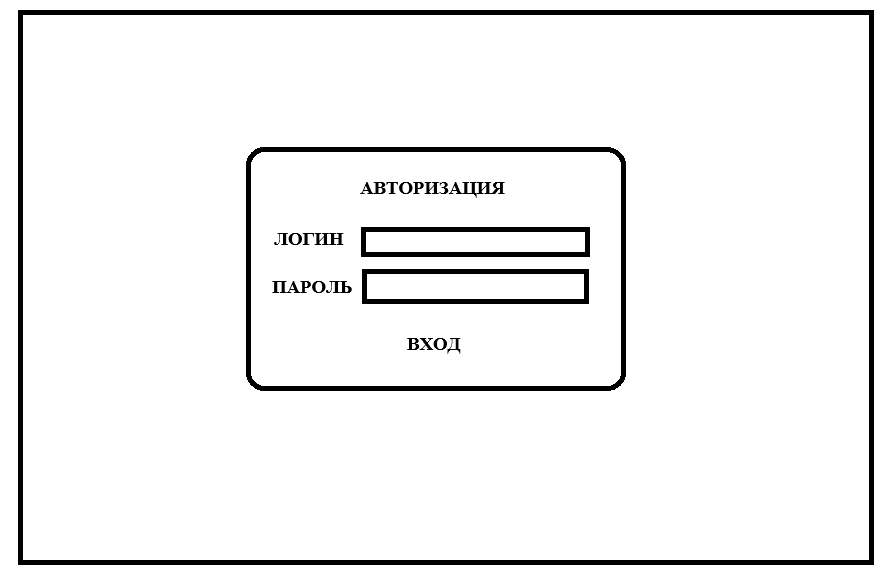


Рисунок 1.5 – Прототип страницы авторизации

# 2. Практическая часть

# 2.1. Назначение и цели создания системы

Полное наименование программного обеспечения и его условное обозначение

Полное наименование – «Разработка приложения «ГИБДД: Universal edition»».

Назначение системы

Автоматизировать фиксацию штрафов и работы со справочной информацией по водителям.

Цели создания приложения:

• Упрощен доступ получения и поиска информация;

• Повышение уровня доступности к предоставленной информации.

# 2.2. Характеристики объекта автоматизации

Краткие сведения об объекте автоматизации

Объекты автоматизации:

• Процесс получения информации о наличии страховки на автомобиль;

• Процесс получения информации о штрафах;

• Процесс получения информации о характеристиках автомобиля;

• Процесс получения информации о водительском удостоверении;

• Процесс получения информации о личных данных пользователя;

В процессе выдачи информации о штрафах, должны отображаться следующие данные:

• Ф.И.О;

• Сущность штрафа;

• Место нарушения;

• Сумма.

Сведения об условиях эксплуатации объекта автоматизации и характеристик окружающей среды

Условия эксплуатации объекта автоматизации (на стороне приложения):

• Операционная система MS Windows 10 или выше;

Условия эксплуатации объекта автоматизации (на стороне сервера БД):

• Операционная система MS Windows 10 или выше;

• SQL Server.

Условия эксплуатации объекта автоматизации (на стороне клиента):

• Операционная система MS Windows 10 или выше;

• SQL Server.

Физические условия эксплуатации и характеристики окружающей среды определяются соответствующими характеристиками окружающей среды используемого аппаратного обеспечения, указанными в документации на него.

# 2.3. Обзор инструментальных средств создания продукта

Для реализации проекта можно воспользоваться следующими средствами:

* Visual Studio – представляет собой полностью интегрированную среду разработки. Она спроектирована таким образом, чтобы делать процесс написания кода, его отладки и компиляции в сборку для поставки конечным потребителям как можно более простым. Является наиболее функционально насыщенной IDE-средой, в которой можно разрабатывать приложения на C#. Данный продукт поставляется с конструкторами графических пользовательских интерфейсов, с возможностями работы с базами данных, с поддержкой фрагментов кода, с возможностями для просмотра всего проекта в целом, с просмотром свойств объектов.
* Standard Widget Toolkit – библиотека графических пользовательских элементов с открытым исходным кодом для разработки интерфейсов пользователя на языке Java. Предоставляет из себя переносимый API для разработчиков Java. Использует собственные библиотеки операционной системы через собственный интерфейс Java (Java Native Interface) для визуализации элементов графического интерфейса, аналогично программам, написанным с использованием собственного API операционной системы.
* PyCharm – это интегрированная среда разработки на языке Python, которая была разработана международной компанией JetBrains (да, и снова эти ребята). Эта IDE распространяется под несколькими лицензиями, в том числе как Community Edition, где чуть урезан функционал. Сами разработчики характеризуют свой продукт как «самую интеллектуальную Python IDE с полным набором средств для эффективной разработки на языке Python», но она не ограничивается на Python. PyCharm поддерживает Google App Engine; IronPython, Jython, Cython, PyPy wxPython, PyQt, PyGTK и др.
* Qt Creator – кроссплатформенная свободная IDE для разработки на С, С++ и QML с помощью фреймворка Qt. Разработана Trolltech для работы с фреймворком Qt. Включает в себя графический интерфейс отладчика и визуальные средства разработки интерфейса как с использованием QtWidgets, так и QML. Зачастую используется при написания программного обеспечения для роботов, автомобилей и кораблей, которые используются на производствах, в перевозке грузов и других похожих отраслях. Qt Creator позволяет добиться высокой производительности при прототипировании интерфейса в приложениях любой сложности.
* CLion – продукт JetBrains. Многофункциональная IDE. В которой поддерживается разработка кода на C и C++, а также web-разработка на HTML(5), CSS, JavaScript, XML. А также на некоторых других языках доступнен в виде плагинов (например, Lua). Основным преимуществом является статический и динамический анализ с безопасный рефакторингом кода. Позволяет кастомизировать редактор так, как удобно пользователю.
* IntelliJ IDEA – ещё один продукт от JetBrains. Комплекс программных средств, который используется для написания, исполнения, отладки и оптимизации кода для многих языков программирования, в частности Java, JavaScript, Python и других языков программирования от компании JetBrains. Отличается обширным набором инструментов для рефакторинга (перепроектирования) и оптимизации кода. Интеграция с серверами приложений, включая Tomcat, TomEE, GlassFish и другие. Имеет набор инструментов для работы с базами данных, SQL файлами и инструменты для запуска тестов и анализа покрытия кода, включая поддержку всех популярных фреймворков для тестирования.
* Eclipse – то бесплатная опенсорсная среда разработки, которая хорошо подойдет как новичкам, так и опытным разработчикам. Помимо инструментов отладки и поддержки Git/CVS, Eclipse поставляется с Java и инструментом для создания плагинов. Изначально Eclipse использовалась только для Java, но сейчас, благодаря плагинам и расширениям, ее функции значительно расширились. Именно из-за возможности расширить Eclipse своими модулями эта платформа и завоевала свою популярность среди разработчиков. Функционал Eclipse не такой большой, как у IntelliJ IDEA, зато эта среда разработки распространяется с открытым исходным кодом. Основными преимуществами является возможность программировать сразу на нескольких языках и возможность интеграции JUnit.

# 2.4. Обоснование выбора средств реализации ИС

Для разработки программного обеспечения мной было принято решение реализовывать информационную системы с нуля в среде Visual Studio 2019 на платформе Windows Presentation Foundation (WPF) используя язык C#. Ниже описано каждое средство реализации, вместе с их преимуществами среди других.

Среда разработки: Visual Studio

* Поддержка контроля версий GIT;
* Технология автодополнения IntelliSense, которая умеет исправлять ошибки разработчика, предлагая ряд вариантов, но, в отдельных случаях, может и сгенерировать фрагменты кода;
* Расширенный функционал, который можно дополнятся плагинами;
* Интегрированная среда разработки (IDE);
* Удобный и интуитивно понятный интерфейс, который можно кастомизировать под каждого пользователя индивидуально;
* Присутствует встроенный отладчик;
* Наличие встроенного средства для быстрого рефакторинга;
* Множество функций, которые позволяют изменять редактор в соответствии с использованием, что означает, что пользователь может загружать библиотеки из Интернета и интегрировать его с кодом в соответствии со своими требованиями.
* Автоматизированный процесс публикации и автоматизации сборки проекта и приложения;
* Наличие множество утилит, который упрощают работу с сторонними компонентами и приложениями, например таким как SQL Server.

Управление базами данных: SQL Server

* Обеспечивает практически неограниченный рост объемов хранения за счет увеличения надежности и масштабируемости системы, используя все преимущества мультипроцессорной обработки файлов;
* Возможность взаимодействия с публичным облаком Windows Azure;
* Облачное резервирование локальных систем и аварийное восстановление из облака повышает отказоустойчивость;
* Возможность интеграция структурированных и неструктурированных данных;
* Наличие интегрированная среды для управления любой инфраструктурой SQL.

# 2.5 Описание логики функционирования ПО

Таблица «Owner» – владельцы автомобилей в приложении и статус пользователя. Данные таблицы «Owner» рисунок 2.1

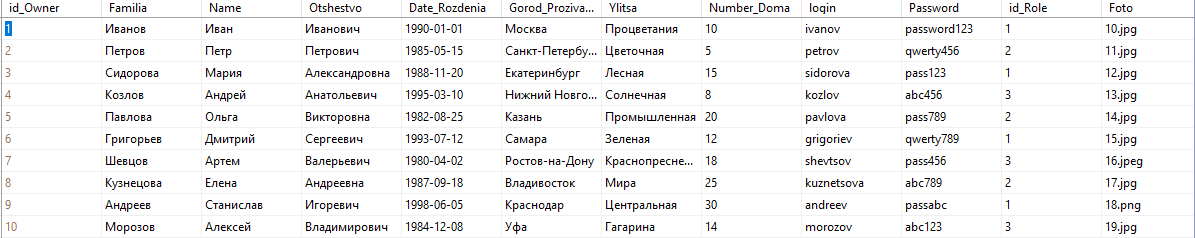


Рисунок 2.1 – Данные таблицы «Owner»

Таблица 2.1 Описание структуры «Owner»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| id\_Owner | int | Ид владельца |
| Familia | nvarchar(50) | Фамилия владельца |
| Name | nvarchar(50) | Имя владельца |
| Otshestvo | nvarchar(50) | Отчество владельца |
| Date\_Rozdenia | date | Дата рождения владельца |
| Gorod\_Prozivania | nvarchar(50) | Город проживания |
| Ylitsa | nvarchar(50) | Улица |
| Number\_Doma | int | Номер дома |
| login | nvarchar(50) | Логин пользователя |
| Password | nvarchar(50) | Пороль пользователя |
| id\_Role | int | Ид роли в системе |
| Foto | nvarchar(50) | Фото владельца |
| Number\_Prav | int | Номер прав |
| Id\_Strahovka | int | Ид страховки |

Таблица «Strahovka» - Данные по оформленным страховым документам. Данные таблицы «Strahovka» рисунок 2.2.

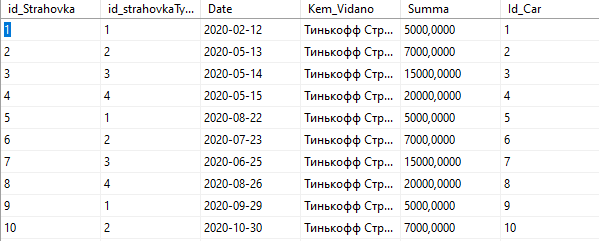


Рисунок 2.2. Данные таблицы Strahovka

Таблица 2.2 Описание структуры «Strahovka»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| id\_Strahovka | int | Ид страховки |
| id\_strahovkaType | int | Ид типа страховки |
| Date | date | Дата оформления страховки |
| Kem\_Vidano | nvarchar(100) | Кем выдана страховка |
| Summa | money | Итоговая сумма |
| Id\_Car | int | Ид машины на которую оформлена страховка |

Таблица «Cars» - Данные по автомобилям пользователей. Данные таблицы «Cars» рисунок 2.3.

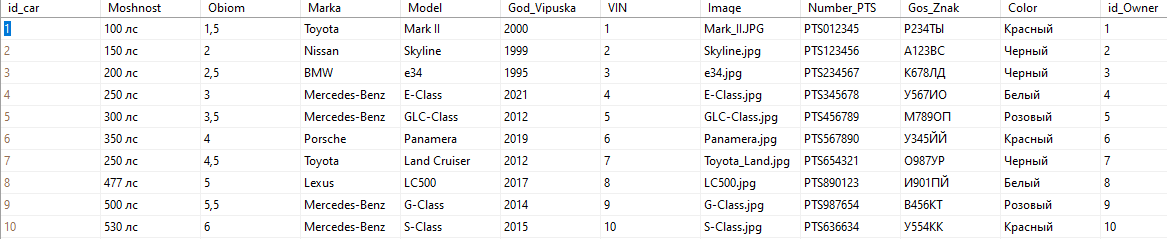


Рисунок 2.3 Данные таблицы «Cars»

Таблица 2.3 Описание структуры «Cars»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| id\_car | int | Ид автомобиля |
| Moshnost | nvarchar(50) | Мощность автомобиля |
| Obiom | nvarchar(50) | Объём двигателя |
| Marka | nvarchar(50) | Марка автомобиля |
| Model | nvarchar(50) | Модель автомобиля |

Продолжение таблицы 2.3

|  |  |  |
| --- | --- | --- |
| God\_Vipuska | date | Год выпуска автомобиля |
| VIN | nvarchar(50) | VIN номер |
| Image | nvarchar(50) | Изображение автомобиля |
| Number\_PTS | nvarchar(50) | Номер ПТС |
| Gos\_Znak | nvarchar(50) | Государственный знак |
| Color | nvarchar(50) | Цвет кузова |
| id\_Owner | int | Ид владельца |

Таблица «Shtraf\_Protokol» - Данные по выписанным штрафам. Данные таблицы «Shtraf\_Protokol» рисунок 2.4.

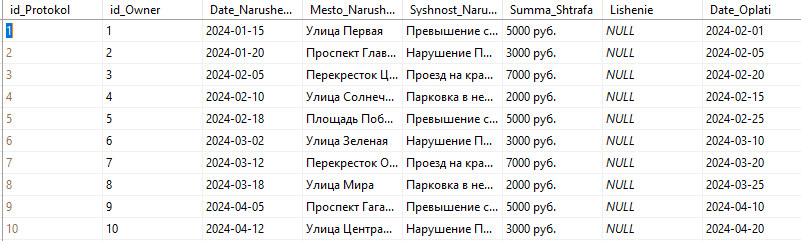


Рисунок 2.4. Данные таблицы «Shtraf\_Protokol»

Таблица 2.4 Описание структуры «Shtraf\_Protokol»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| id\_Protokol | int | Ид протокола |
| id\_Owner | int | Ид владельца |
| Date\_Narushenia | date | Дата нарушения |
| Mesto\_Narushenia | nvarchar(50) | Место нарушения |
| Syshnost\_Narushenia | nvarchar(MAX) | Сущность нарушения |

Продолжение таблицы 2.4

|  |  |  |
| --- | --- | --- |
| Summa\_Shtrafa | nvarchar(50) | Сумма штрафа |
| Lishenie | nvarchar(50) | Лишение прав |
| Date\_Oplati | date | Дата оплаты |

Таблица «Num\_Prav» - Данные по водительским удостоверениям. Данные таблицы «Num\_Prav» рисунок 2.5.

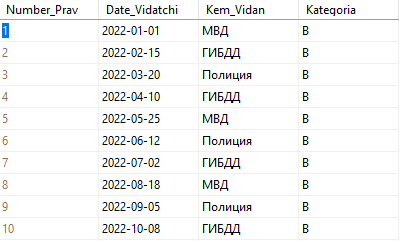


Рисунок 2.5. Данные таблицы «Num\_Prav»

Таблица 2.5 Описание структуры «Shtraf\_Protokol»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| Number\_Prav | int | Номер прав |
| Date\_Vidatchi | date | Дата выдачи |
| Kem\_Vidan | nvarchar(50) | Кем выдано |
| Kategoria | nvarchar(50) | Категория прав |

Таблица «Strahovka\_Type» - Данные по типам страховых сертификатов. Данные таблицы «Strahovka\_Type» рисунок 2.6.

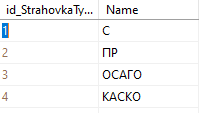


Рисунок 2.6. Данные таблицы «Strahovka\_Type»

Таблица 2.6 Описание структуры «Shtraf\_Protokol»

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| id\_StrahovkaType | int | Ид типа страховки |
| Name | nvarchar(50) | Наименование |

Таблица «Role» - Данные по типам страховых сертификатов. Данные таблицы «Role» рисунок 2.7.

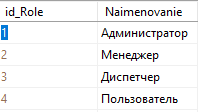


Рисунок 2.7. Данные таблицы «Role»

Таблица 2.7 Описание структуры «Shtraf\_Protokol»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| id\_Role | int | Ид Роли |
| Naimenovanie | nvarchar(50) | Наименование роли |

# 2.6 Реализация интерфейса ПО

Интерфейс программы является оконным и содержит несколько окон как для пользователя, так и для администратора.

В зависимости от критериев запуска приложения открываются разные окна, если приложение открыто от имени администратора, тогда открывается окно специалиста, где может взаимодействовать с оборудованием и заявками. В ином случае открывается окно пользователя.

Разработка логотипа – вектор/прорисовка/интегрирование в приложение



Рисунок 2.8. Логотип приложения

Выбор цветовой модели (выбор палитры/цветового круга/треугольника)

Таблица 2.8 Цветовая модель приложения

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Основной фон | | Дополнительный фон | | Акцентирование  внимания | |
| RGB  #FFF0F8FF | | RGB  #FFFF0202 | | RGB  #FF340BFF | |
| (240, 248, 255) | | (255, 2, 2) | | (52, 11,255) | |
|  | |  | |  | |

2.6 Разработка/код объектов экрана – кнопки/чек-боксы и др – предоставить код app/css/скрины и скриншот разработанного объекта.

App.xaml зафиксировал в себе значения стилей для объектов в программе. Предоставлено в листинге 2.1

Листинг 2.1

|  |
| --- |
| <Application.Resources> <Style TargetType="TextBlock"> <Setter Property="FontSize" Value="26"/> <Setter Property="FontFamily" Value="Segoe UI"/> </Style> <Style TargetType="GroupBox"> <Setter Property="HeaderTemplate"> <Setter.Value> <DataTemplate> <TextBlock Text="{Binding}" FontWeight="Bold"/> </DataTemplate> </Setter.Value> </Setter> </Style> <Style TargetType="TextBlock" x:Key="TxblTitle"> <Setter Property="HorizontalAlignment" Value="Center"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="FontFamily" Value="Segoe UI"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontSize" Value="30"/> <Setter Property="TextAlignment" Value="Center"/> <Setter Property="Margin" Value="5"/> </Style> <Style TargetType="TextBlock" x:Key="TxblLogPass"> <Setter Property="FontFamily" Value="Segoe UI"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontSize" Value="15"/> <Setter Property="TextAlignment" Value="Right"/> <Setter Property="Width" Value="75"/> <Setter Property="Margin" Value="5"/> </Style> <Style TargetType="TextBox" x:Key="TxbLogin"> <Setter Property="FontFamily" Value="Segoe UI"/> <Setter Property="FontSize" Value="15"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="Width" Value="200"/> </Style> <Style TargetType="PasswordBox" x:Key="PsbPassword"> <Setter Property="FontFamily" Value="Segoe UI"/> <Setter Property="FontSize" Value="15"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="Width" Value="200"/> </Style> </Application.Resources> |

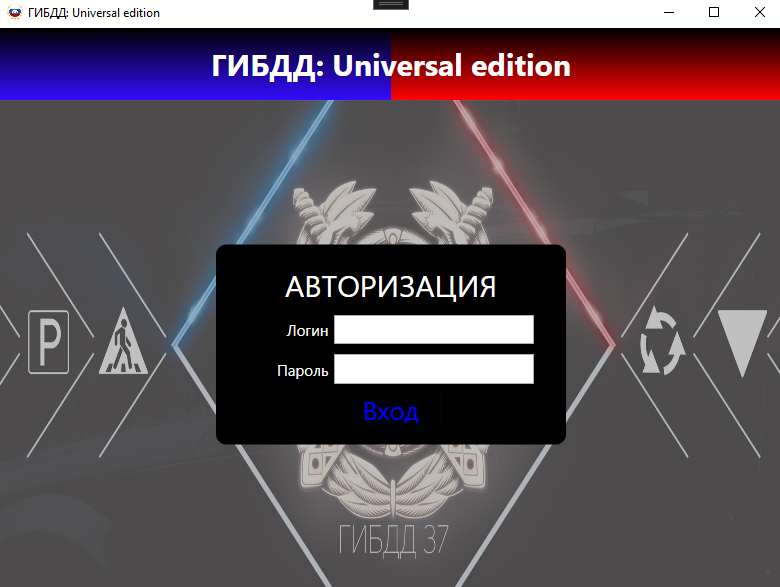
MainPage хранит в себе значения кнопок которые отвечают за показ сертификатов страхования в приложении и имеют одноименные названия дабы сделать более понятным чтение кода. Предоставлено в листинге 2.2

Листинг 2.2

|  |
| --- |
| public MainPage() { InitializeComponent(); DataContext = Manager.currentUser; DrgLb.ItemsSource = Talyzenkov\_GIBDDEntities3.GetContext().Shtraf\_Protokol.Where(x => x.id\_Owner == Manager.currentUser.id\_Owner).ToList(); Drg.ItemsSource = Talyzenkov\_GIBDDEntities3.GetContext().Cars.Where(x => x.id\_Owner == Manager.currentUser.id\_Owner).ToList(); } private void C\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Visible; KASKO.Visibility = Visibility.Hidden; Osago.Visibility = Visibility.Hidden; PR.Visibility = Visibility.Hidden; } private void ПР\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Hidden; KASKO.Visibility = Visibility.Hidden; Osago.Visibility = Visibility.Hidden; PR.Visibility = Visibility.Visible; } private void Каско\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Hidden; KASKO.Visibility = Visibility.Visible; Osago.Visibility = Visibility.Hidden; PR.Visibility = Visibility.Hidden; } private void Осаго\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Hidden; KASKO.Visibility = Visibility.Hidden; Osago.Visibility = Visibility.Visible; PR.Visibility = Visibility.Hidden; } |

Скриншоты итогового приложения с описанием работы – навигация/переходы

При запуске приложения пользователь видит форму авторизации (рисунок 2.9).

Рисунок 2.9 - Форма авторизации

При успешной авторизации пользователь перенаправляется в главную форму раздел «Личные данные» (рисунок 2.10).

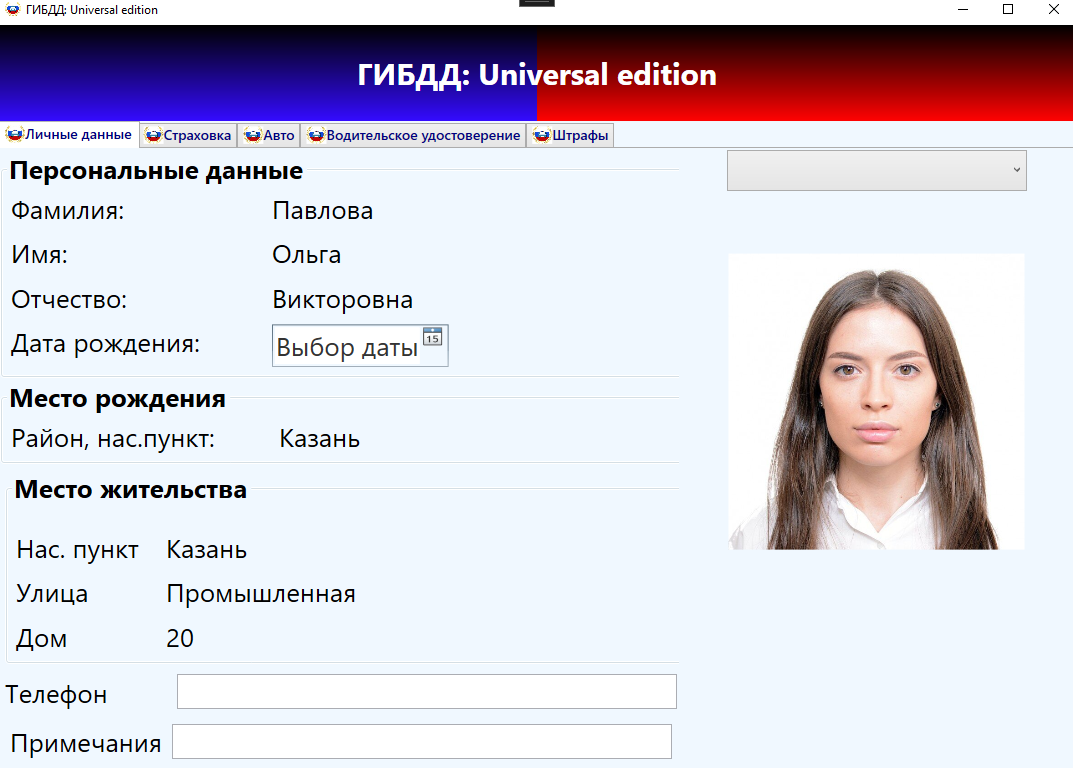


Рисунок 2.10 – Главная форма (Личные данные)

При выборе раздела с наименованием «Страховка» пользователь попадает на страницу где указаны виды страховки, которые при нажатии отображают страховой сертификат (рисунок 2.11).

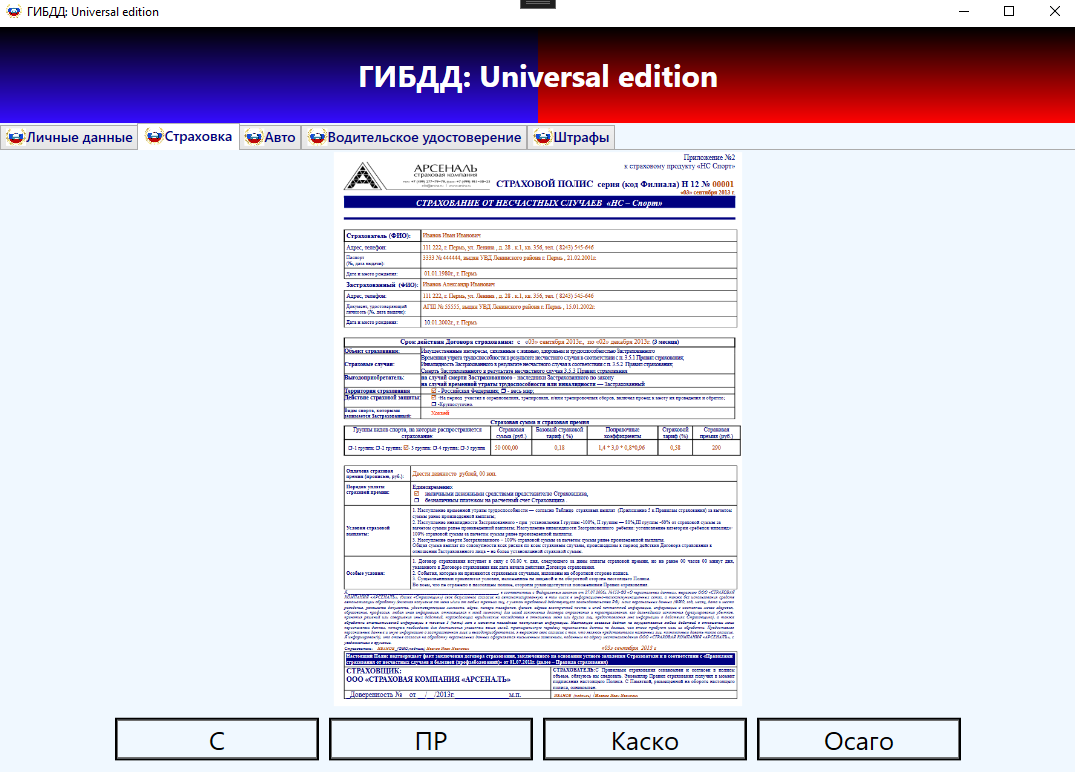


Рисунок 2.11 – Главная форма (Страховка)

При выборе раздела с наименованием «Авто», пользователь попадает в реестр автомобильных данных, в котором указано авто, которое зарегистрировано на авторизированное лицо, заверенное печатью, отображает: ид авто в базе данных, его мощность, объём двигателя, марку, модель, год выпуска и VIN номер автомобиля (рисунок 2.12).



Рисунок 2.12 – Главная форма (Авто)

При выборе раздела с наименованием «Водительское удостоверение», пользователь видит своё водительское удостоверение(рисунок 2.13).



Рисунок 2.13 – Главная форма (Авто)

При выборе раздела с наименованием «Штрафы», пользователь попадает в свою историю штрафов где отображено: Ид протокола, Дата нарушения, Место нарушения, Причина по которой выписан штраф, Сумма денежных средств, которые нарушитель обязан выплатить и Дата оплаты штрафа (рисунок 2.14).

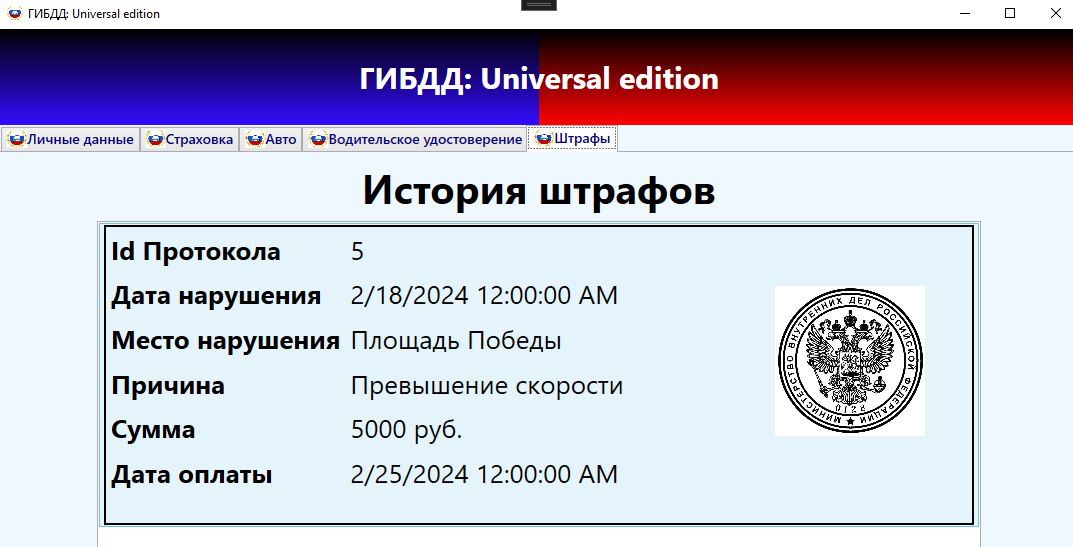


Рисунок 2.14 – Главная форма (Штрафы)

# 2.7. Написание программного кода

Описание разработанных функций/кода – что он выполняет

На главном окне выводит форму авторизации через инструмент Frame

(Листинг 2.3).

Листинг 2.3 MainWindow.cs

|  |
| --- |
| public MainWindow() { InitializeComponent(); Manager.MainFrame = FrameMain; FrameMain.Navigate(new avtoriz()); } |

На основной странице в инициализационном компоненте указаны ссылки откуда берутся данные, для отображения в присвоенных именем полях данные берутся из базы данных и связываются через GetContext. DrgLb отвечает за отображение данных из таблицы Shtraf\_Protokol, которые присваиваются к конкретному пользователю. Drg отвечает за отображение данных с таблицы Cars, которые присвоены авторизированному пользователю (Листинг 2.4.).

|  |
| --- |
| public MainPage() { InitializeComponent(); DataContext = Manager.currentUser; DrgLb.ItemsSource = Talyzenkov\_GIBDDEntities3.GetContext().Shtraf\_Protokol.Where(x => x.id\_Owner == Manager.currentUser.id\_Owner).ToList(); Drg.ItemsSource = Talyzenkov\_GIBDDEntities3.GetContext().Cars.Where(x => x.id\_Owner == Manager.currentUser.id\_Owner).ToList(); } |

Листинг 2.4. – MainPage инициализационный компонент

MainPage хранит в себе значения кнопок которые отвечают за показ сертификатов страхования в приложении и имеют одноименные названия дабы сделать более понятным чтение кода (Листинг 2.5.).

|  |
| --- |
| private void C\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Visible; KASKO.Visibility = Visibility.Hidden; Osago.Visibility = Visibility.Hidden; PR.Visibility = Visibility.Hidden; } private void ПР\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Hidden; KASKO.Visibility = Visibility.Hidden; Osago.Visibility = Visibility.Hidden; PR.Visibility = Visibility.Visible; } private void Каско\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Hidden; KASKO.Visibility = Visibility.Visible; Osago.Visibility = Visibility.Hidden; PR.Visibility = Visibility.Hidden; } private void Осаго\_Click(object sender, RoutedEventArgs e) { Ctrax.Visibility = Visibility.Hidden; KASKO.Visibility = Visibility.Hidden; Osago.Visibility = Visibility.Visible; PR.Visibility = Visibility.Hidden; } |

Листинг 2.5. – MainPage кнопки

# 2.8 Защита информации

Реализация защиты информации – авторизация успешная (рисунок 2.15.).

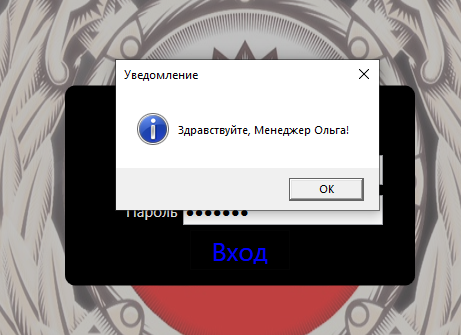


Рисунок 2.15. –Успешная авторизация в приложении

Реализация защиты информации – авторизация ошибочная (рисунок 2.16.).

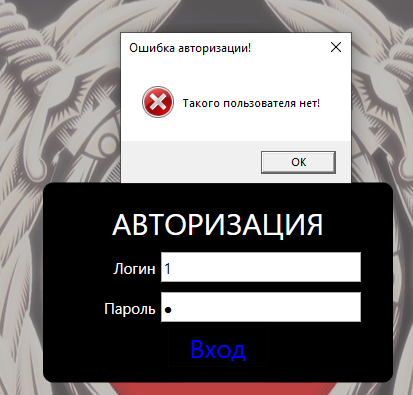


Рисунок 2.16. –Неверные данные при авторизация в приложении

# Заключение

В ходе выполнения курсового проекта был разработан модуль информационной системы для ГИБДД.

В ходе курсового проекта был реализован модуль информационной системы, который позволит упрощении и автоматизация процесса получения информации о состоянии компьютерного оборудования, его распределении, использовании и технической поддержке

Также закреплены теоретические знания и практические навыки работы с SQL Server, C# и XAML, умение работать с литературой, анализировать источники, делать обоснованные выводы.

Полученные теоретические и практические навыки будут использованы в дальнейшем обучении и также в написании дипломной работы в будущем. Провел предварительное планирование и проверку осуществимости.

В рамках курсового проекта были выполнены следующие цели

• Упрощен доступ получения и поиска информация;

• Повышение уровня доступности к предоставленной информации относительно установленного оборудования.

А также поставленные задачи:

1. Определил требования к программному продукту.
2. Разработал прототип приложения.
3. Реализовал приложение в выбранной среде IDE.
4. Создал сборку для установки.
5. Разместил проект на веб-сервисе GitHub или аналогичном – https://github.com/fxxkclown/GIBDD\_Talyzenkov.git

# Список использованных источников

1. Казанский, А. А. Программирование на Visual C# : учебное пособие для среднего профессионального образования / А. А. Казанский. — 2-е изд., переработан и дополнен — Москва: Издательство Юрайт, 2024. — 192 с. — (Профессиональное образование). — ISBN 978-5-534-14130-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/538155

2. Тузовский, А. Ф. Объектно-ориентированное программирование: учебное пособие для вузов / А. Ф. Тузовский. — Москва: Издательство Юрайт, 2024. — 213 с. — (Высшее образование). — ISBN 978-5-534-16316-2. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/537332

3. Казанский, А. А. Программирование на Visual C#: учебное пособие для вузов / А. А. Казанский. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2024. — 192 с. — (Высшее образование). — ISBN 978-5-534-12338-8. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/537364

4. Демин, А. Ю. Информатика. Лабораторный практикум: учебное пособие для вузов / А. Ю. Демин, В. А. Дорофеев. — Москва: Издательство Юрайт, 2024. — 147 с. — (Высшее образование). — ISBN 978-5-534-18259-0. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/534628

5. Сысолетин, Е. Г. Разработка интернет-приложений: учебное пособие для вузов / Е. Г. Сысолетин, С. Д. Ростунцев; под научной редакцией Л. Г. Доронинского. — Москва: Издательство Юрайт, 2022. — 90 с. — (Высшее образование). — ISBN 978-5-9916-9975-4. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/