Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Дисциплина «Структуры и базы данных»

|  |  |
| --- | --- |
|  | «К ЗАЩИТЕ ДОПУСТИТЬ» |
|  | Руководитель курсового проекта  канд.техн.наук, доцент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*\*\*\*\*\*\*\*\*. |
|  | \_\_\_.\_\_\_\_.2022 |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту

на тему:

**«БАЗА ДАННЫХ ДЛЯ ПОДДЕРЖКИ работы ГАИ»**

БГУИР КП 1-39 03 02 \*\*\* ПЗ

|  |  |
| --- | --- |
|  | Выполнил студент группы \*\*\*\*\*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |
|  | Курсовой проект представлен на проверку \_\_\_.\_\_\_\_.2022  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |

Минск 2022

**РЕФЕРАТ**

БГУИР КП 1-39 03 02 \*\*\* ПЗ

**\*\*\*\*\*\*\*\*.** База данных для поддержки работы ГАИ: пояснительная записка к курсовому проекту / \*\*\*\*\*\*\* – Минск: БГУИР, 2022. – 30 с.

Пояснительная записка 30 с., 14 рис., 11 источников, 4 приложения, 5 графических материалов.

АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ЕЁ ФОРМАЛИЗАЦИИ ДЛЯ ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ, ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ ДЛЯ ОСНОВНОГО ВИДА ДЕЯТЕЛЬНОСТИ РАССМАТРИВАЕОМЙ ОБЛАСТИ, ПРИМЕНЕНИЕ РАЗРАБОТАННОЙ БАЗЫ ДАННЫХ

*Цель* *проектирования*: разработка *WEB*-приложения, проектирование эффективной и безопасной базы данных для поддержания работы ГАИ.

*Методология проведения работы*: В процессе решения поставленных задач использованы принципы системного подхода, теория проектирования базы данных и методы их связки, методы и паттерны проектирования *Web* -приложения.

*Результаты работы*: изучены способы хранения информации в базе данных, проведен анализ реляционной модели данных, разработано *Web*-приложение для обеспечения работы ГАИ.

*Область применения результатов*: разработанная база данных и *Web*-приложение в будущем, может использоваться для создания полноценного приложения для обеспечения комфортного использования базы данных ГАИ.

**СОДЕРЖАНИЕ**

[Введение](#_Toc65079205) 5

[1 Анализ предметной](#_Toc65079206) области и ее формализация для проектирования базы данных .6

[1.1 Описание предметной области](#_Toc65079207) 7

1.2 Анализ информационных потребностей пользователей

и предварительное описание запросов 7

1.3 Определение требований и ограничений к базе данных с точки зрения предметной области 8

1.4 Постоновка решаемой задачи 9

[2 Проекти](#_Toc65079208)рование базы данных для основного вида деятельности

рассматриваемой предметной области 10

[2.1 Разработка инфологической модели предметной области](#_Toc65079209) базы данных 10

[2.2 Выбор и обоснование используемых типов данных и ограничений](#_Toc65079210)

(доменов) 12

[2.3 Проектирование за](#_Toc65079211)просов к базе данных 13

[2.4 Програмная](#_Toc65079212) реализация и документирование базы данных .14

[3 При](#_Toc65079214)менение разработанной базы данных .15

[3.](#_Toc65079211)1 Руководство пользователя 15

[3.](#_Toc65079211)2 Администрирование базы данных 18

[3.](#_Toc65079211)3 Реализация клиентских запросов 20

[3.](#_Toc65079212)4 Обоснование и реализация механизма обеспечения безопасности

и сохранности данных .21

[Заключение](#_Toc65079217) 23

[Список использованных источников](#_Toc65079218) 24

[Приложение А (обязательное) Скрипт генерации БД](#_Toc65079219) 25

[Приложение Б (обязательное) Листинг программного кода](#_Toc65079222) 27

Приложение В (обязательное) Отчёт о проверке на взаимствования

в системе «Антиплагиат» 29

Приложение Г (обязательное) [Ведомость курсового проекта](#_Toc65079222) 30

**ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ**

**И ТЕРМИНОВ**

|  |  |
| --- | --- |
| БГУИР | – Белорусский Государственный Университет Информатики и Радиоэлектроники; |
| КП | – курсовой проект; |
| ГАИ | – Государственная автомобильная инспекция; |
| ПЗ | – пояснительная записка; |
| ПИКС | – кафедра проектирования информационно-компьютерных систем; |
| СССР | – Союз Советских Социалистических Республик; |
| ГАИ | – Государственная автомобильная инспекция; |
| СУБД | – система управления базами данных; |
| БД | – база данных; |
| ФИО | – Фамилия, Имя, Отчество; |

|  |  |
| --- | --- |
| *VIN* | – *Vehicle Identification Number* |
| *API* | – *application programming interface*; |
| НФ | – нормальная форма; |
| *HTML* | – *HyperText Markup Language*; |
| *CSS* | – каскадные таблицы стилей, формальный язык описания внешнего вида документа; |
| *HTTP* | – *HyperText Transfer Protocol*; |
| *HTTPS* | – *HyperText Transfer Protocol Secure*; |
| *SSL* | – *Secure Sockets*; |
| *SQL* | – *structured query language*; |
| *TLS* | – *transport layer security;* |
| *URL* | – *Uniform Resource Locator*; |
| *ROR* | – *Ruby on Rails.* |

ВВЕДЕНИЕ

3 июля 1936 года постановлением Совнаркома СССР № 1182 было утверждено Положение о Государственной автомобильной инспекции Главного управления Рабоче-крестьянской милиции Народного комиссариата внутренних дел СССР. Белорусская Госавтоинспекция ведет свое летоисчисление именно с этой даты.

Ее тогдашние установки предопределили основные направления деятельности ГАИ. На Госавтоинспекцию возлагались обязанности по учету дорожно-транспортных происшествий, выявлению и анализу их причин, привлечению к ответственности виновных лиц; по выдаче регистрационных знаков и технических паспортов, периодическому техническому осмотру автомобилей, мотоциклов и автобусов; по учету транспортных средств.

На тот момент служба не имела вычислительной техники, которая могла создавать и обрабатывать базы данных, поэтому сотрудники взаимодействовали с бумажными носителями такими как справочники, документы с записями о водителях и транспортных средствах. Такую форму записи можно считать бумажной базой данных. Но она имеет ряд ограничений и недостатков, с которыми приходилось мириться, например: непоследовательность и невозможность сортировки данных, дублирование данных, данные легко потерять или намеренно уничтожить, трудно редактировать записи.

Но с тех пор, как наступил век информационных технологий, сотрудники ГАИ используют компьютер и имеют всю информацию в электронном виде. Также использование информационных баз данных дало возможность гражданам использовать их для получение критической важной информации об авто (данные о розыске, ограничения на использование авто, утраченных регистрационных знаки).

Объектом исследования данной курсовой работы является структура Госавтоинспекции.

Цель исследования – изучение предметной области и разработка модели базы данных для обеспечения целостного функционирования Госавтоинспекции.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ЕЕ ФОРМАЛИЗАЦИЯ ДЛЯ ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ

Для того чтобы понять, что является предметной областью, нужно выяснить, что это такое. На основе полученных данных начинать проектирование.

Предметная область – это часть реального мира, данные о котором необходимо отразить в базе данных. Исходя, из определения анализом предметной области состоит из ее подробного описания, выделения полезной информации и последующей формализации в данные.

В результате формализации информации о предметной области мы получаем данные в виде сущностей и их атрибутов [1].

**1.1 Описание предметной области**

В современном мире с каждым днём всё острее стоит вопрос информатизации и компьютеризации. С помочью компьютеров доступ к любого рода информации и её последующей обработке проще осуществить.

Информатизации Госавтоинспекции, в том числе проектирование и обработка базы данных для ГАИ является актуальной проблемой, т.к. ГАИ использует огромный объём разнообразной и постоянно обновляющей информации.

ГАИ представляет собой такую организация, где наличие и функционирование базы данных и своевременное обновление информации являются одними из главных критериев для качественной и быстро работы. Главным пользователем базы данных будет инспектор ГАИ, исходя из этого можно определить главные задачи разрабатываемого программного обеспечения:

– полная информация о транспортном средстве и водителе;

– быстрый доступ к информации;

После детального изучения и анализа предметной области будет разработана структура базы данных для поддержки работы ГАИ. Предметная область, содержит полную информацию о транспортном средстве и водителе.

Важной деталью является то, что разным инспекторам нужна разная информация о транспортном средстве и водителе. Поэтому база данных будет разделена на несколько удобных таблиц.

**1.2 Анализ информационных потребностей пользователей, предварительное описание запросов**

В данной главе будут определены потенциальные пользователи базы данных, а также сформулированы ограничения и привилегии, которые будут присваиваться каждой группе, на предоставление, внесение, обновление, изменение и удаление данных.

Потенциальные пользователи данной базы данных – это инспектора ГАИ и обычные граждане. Для инспектора база данных основное средство работы, поэтому он должен иметь наиболее полную информацию в зависимости от конкретного отдела его работы.

Инспектор, в зависимости от отдела, где он работает должен получить информацию об авто (первая дата регистрация, дата постановки на учёт, текущий гос. номер, *VIN* номер, информация о штрафах, информация о розыске), а также информацию о водителе (имя, фамилия, дата рождения, дата получения прав). Обычный же гражданин сможет получить информацию о розыске авто, ограничениях и информацию о потерянных номерах.

Очевидно, что обычный гражданин не может напрямую внести информацию в базу данных. Инспектор может добавлять новые строки в базу данных, а также изменять старые. Удалять данные из базы данных может только начальник местного отделения ГАИ с одобрения главного управление ГАИ или с постановления суда, формально доступ к удалению строк есть ещё и у администратора БД, но пользоваться он этим не должен. Таблица 1.1 наглядно отображает пользователей базы данных и их права.

Таблица 1.1 – Пользователи базы данных и их привилегии

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Пользователи/Привилегии | Обычный гражданин | Инспектор ГАИ | Начальник ГАИ | Администратор БД |
| Предоставление | + | + | + | + |
| Внесение | - | + | + | + |
| Обновление | - | + | + | + |
| Изменение | - | + | + | + |
| Удаление | - | - | + | + |

**1.3 Определение требований и ограничений к базе данных с точки**

**зрения предметной области**

База данных представляет собой совокупность структурированных взаимосвязанных данных, относящихся к определённой предметной области и организованных для решения задач разными пользователями.

Исходя из выбранной предметной области были выявлены следующие требования:

− *VIN* номер у каждой машины должен быть уникальным и иметь длину строго равную 22 символам;

− у машины обязательно должен быть владелец;

− вход в базу данных под уникальным идентификатором;

− без входа в личный кабинет сайт предоставляет ограниченный спектр возможностей считая пользователя обычным гражданином;

– каждое поле базы данных должно быть проверено перед занесением в базу данных;

– запись в таблице «Жалобы на авто» не может быть создана, без записи в таблице «Регистрации»;

– запись в таблице «Штрафы» не может быть создана, без записи в таблице «Регистрации»;

– запись в таблице «Регистрации» не может быть создана, без записи в таблицах «Владельцы авто», «Информация о машине»;

– *id* каждой записи в любой из баз данных, не может повторяться внутри базы или равняться *null* и должно быть строго положительным;

– записи должны удалять в двух таблицах, если между ними имеется связь;

– таблицы не должны иметь повторения данных, то есть должна быть приведена минимум к 1НФ;

– при некорректном занесении данных в таблицу, должно появляться поле с явным указанием ошибок, которые нужно исправить;

– имя базы должно отражать её функционал;

– база данных не должна содержать лишних полей.

Подробнее разберём пункт «каждое поле базы данных должно быть проверено перед занесением в базу данных», данный пункт означает, что каждого каждое поле должно валидироваться перед внесением его в базу данных, например: в базу данных «Регистрации», не должен попасть номер «AA99-B» или «ББОЛ-АМ6», в данном поле будет осуществлять валидация по регулярному выражению /*[A-Z]{4}[A-Z]{2}-[1-7]*/, которое позволить не заносить в базу данных некорректные требования.

**1.4 Постановка решаемой задачи**

Целью курсового проекта является разработка базы данных для поддержки работы ГАИ. С помощью базы данных должны быть обеспечены требования надежности, минимальной избыточности, целостности данных и ее схема должна быть приведена к третьей нормальной форме. В дополнение к этому она должна поддерживать современные средства для работы с данными.

Вся информация представлена в виде взаимосвязанных между собой таблиц, которые состоят из строк и столбцов. Каждый документ должен иметь первичный ключ, который однозначно определяет запись в таблице и отличает ее от других [2]. При добавлении поля, содержащего значение первичного ключа второй таблицы в первую таблицу образуется связь между двумя таблицами.

В ходе проектирования курсового задания были выявлены следующие цели:

− способность получить наиболее актуальную информацию об авто или автовладельце;

− возможность удобного и быстрого администрирования базы данных;

Для решения вышеперечисленных целей были сформулированы следующие задача:

− разработка базы данных, в которой не будет находиться избыточная информация, для её быстродействия;

Таким образом будет спроектирована безотказная и удобная БД для обеспечения работы Госавтоинспекции.

# 2 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ ДЛЯ ОСНОВНОГО ВИДА ДЕТЕЛЬНОСТИ РАССМАТРИВАЕМОЙ ПРЕДМЕТНОЙ ОБЛАСТИ

Проектирование базы данных – сложная и комплексная задача, связанная с созданием информационной системы. На этапе проектирования надо создать максимально подходящую БД, потому что последующие изменения приведут к большим финансовым потерям. В этой главе определим содержание базы данных, способ организации данных и инструментальные средства управления данными.

**2.1 Разработка инфологической модели предметной области базы** **данных**

Цель инфологического моделирования – обеспечение наиболее естественных для человека способов сбора и представления той информации, которую предполагается хранить в создаваемой базе данных. Храниться данные будут в диаграмме «сущность – связь», которая будет создана с помощью системы управления базами данных [1].

Связь – это некоторая ассоциация между двумя сущностями. Одна сущность может быть связана с другой или сама с собою [1].

Диаграмма «сущность – связь» (*ER*-диаграмма) позволяет графически представить все элементы информационной модели согласно простым, интуитивно понятным, но строго определенным правилам – нотациям [1].

В соответствии с индивидуальным заданием были определены требования к составу данных. На основе анализа предметной области выделены следующие сущности:

– сущность «Информация о владельце»;

– сущность «Информация о регистрации»;

– сущность «Информация о машине»;

– сущность «Штрафы»;

– сущность «Жалобы»;

– сущность «Розыск»;

*ER*-диаграмма представления базы данных приведена на рисунке 2.1.

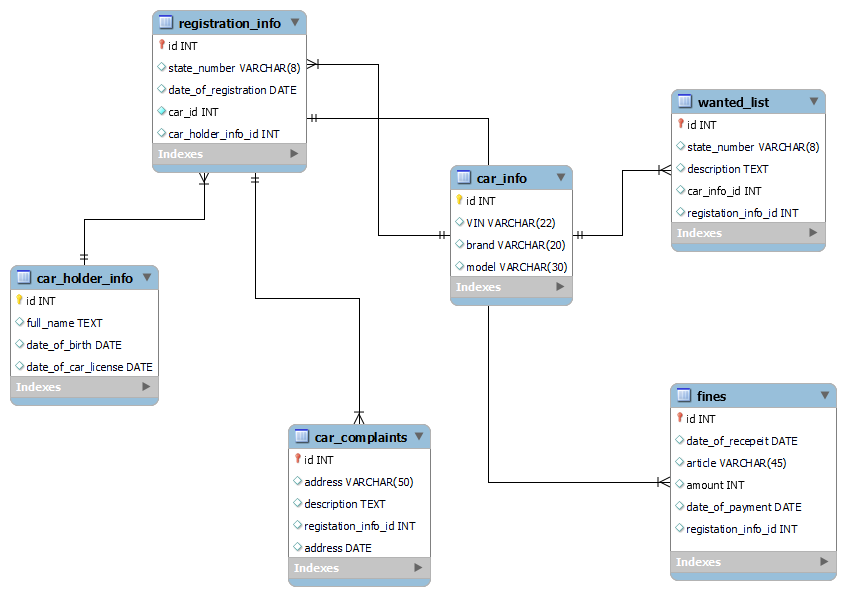


Рисунок 2.1 – *ER*-диаграмма базы данных

Первичный ключ представляет собой один из примеров уникального индекса и применяется для уникальной идентификации записей таблицы. Никакие две записи таблицы не могут иметь одинаковые значения первичного ключа. Первичный ключ обычно сокращенно обозначают как *PK (Primary key)*. Название первичного ключа должно быть уникальным и не совпадать ни с одним атрибутом других сущностей, а также со структурами запросов и названием привилегий.

В реляционных базах данных практически всегда разные таблицы логически связаны друг с другом. Первичные ключи как раз используются для однозначной организации такой связи.

Внешние ключи используются главным образом для проверки целостности данных, а не для объединения таблиц, как принято считать, однако в нашем случае внешний ключ будет применяться для связи между таблицами [1].

Внешние ключи позволяют установить связи между таблицами. Внешний ключ устанавливается для столбцов из зависимой, подчиненной таблицы, и указывает на один из столбцов из главной таблицы. Как правило, внешний ключ указывает на первичный ключ из связанной главной таблицы [2].

**Нормализация** – это процесс удаления избыточных данных. Также нормализацию можно рассматривать и с позиции проектирования базы данных, в таком случае мы можем сформулировать определение нормализации следующим образом. избыточность данных создает предпосылки для появления различных аномалий, снижает производительность, и делает управление данными не гибким и не очень удобным. Отсюда можно сделать вывод, что нормализация нужна для: устранения аномалий, повышения производительности, повышения удобства управления данными. Избыточность данных – это когда одни и те же данные хранятся в базе в нескольких местах, именно это и приводит к аномалиям [3].

При работе с реляционными базами данных обязательным является удовлетворение только требованиям первой нормальной формы (1НФ). Нормальная форма – требование, предъявляемое к структуре таблиц в теории реляционных баз данных для устранения из базы избыточных функциональных зависимостей между атрибутами (полями таблиц).

Первая нормальная форма – все атрибуты являются простыми, все используемые домены должны содержать только скалярные значения, не должно быть повторение строк в таблице.

Вторая нормальная форма – выполняются все требования 1НФ и каждый не ключевой атрибут неприводимо зависит от *Primary key*.

Третья нормальная форма – выполняются все требования 2НФ и каждый неключевой атрибут нетранзитивно зависит от первичного ключа. Иначе говоря, все поля, которые могут относится к многим таблицам выносятся в отдельные таблицы [4].

**2.2 Выбор и обоснование используемых типов данных, и ограничений (доменов)**

В *MySQL* используется множество типов данных, которые можно разделить на три основные группы:

– числовые;

– дата и время;

– строковые.

В проектируемой базе данных были использованы:

– *INT* тип данных для хранения чисел;

– *VARCHAR* текстовое поле переменной длины, для названий;

– *TEXT* представление информации строкового вида;

– *DATETIME* для хранения даты и время.

Есть несколько правил, которые определяют отношения между таблицами:

– отношение 1:1 (один к одному): первичный ключ для одной из таблиц включен в качестве внешнего ключа в другой таблице;

– отношение 1:n (один ко многим): первичный ключ из таблицы 1 добавляется в качестве внешнего ключа в таблицу n;

– отношение n:m (многие ко многим): создается новая таблица (таблица связи), первичный ключ состоит из первичных ключей двух оригинальных таблиц.

В следующем разделе рассматриваются запросы к базе данных от пользователей.

**2.3 Проектирование запросов к базе данных**

В данном курсовом проекте, база данных содержит 6 логических и физических баз данных. Были реализованы операции добавления, удаления, просмотра и обновления информации из базы данных. В *Ruby on Rails* запросы к базе данных можно делать по средствам языка без использования *SQL* с помощью использования дополнения *Active Record* (установленного по умолчанию).

*Active Record* это *M* в *MVC* – модель – которая является слоем в системе, ответственным за представление бизнес-логики и данных. *Active Record* упрощает создание и использование бизнес-объектов, данные которых требуют персистентного хранения в базе данных. Сама по себе эта реализация паттерна *Active Record* является описанием системы *ORM*.

Класс модели – единственное число с первой прописной буквой в каждом слове. Таблица базы данных – множественная форма со словами, разделенными знаком подчеркивания.

*Active Record* использует соглашения о именовании для столбцов в таблицах базы данных, зависящих от назначения этих столбцов**. Внешние ключи** – эти поля должны именоваться по образцу *singularized\_table\_name\_id* (т.е., *item\_id*, *order\_id*). Это поля, которые ищет Active Record при создании связей между вашими моделями. **Первичные ключи** - По умолчанию Active Record использует числовой столбец с именем *id* как первичный ключ таблицы (*bigint* для *PostgreSQL* и *MySQL*, *integer* для *SQLite*). Этот столбец будет автоматически создан при использовании миграций *Active Record* для создания таблиц [5].

**2.4 Программная реализация и документирование базы данных**

Программная реализация приложения будет использовать *Ruby on Rails, HTML5, CSS, JavaScript, Bootstrap 5,* для серверной и клиентской части. *Ruby on Rails* фреймворк, написанный на языке программирования Ruby, реализует архитектурный шаблон *Model-View-Controller* для веб-приложений, а также обеспечивает их интеграцию с веб-сервером и сервером баз данных. Является открытым программным обеспечением и распространяется под лицензией *MIT* [6].

*HTML5* – язык для структурирования и представления содержимого всемирной паутины. Это пятая версия *HTML*. Цель разработки *HTML5* – улучшение уровня поддержки мультимедиатехнологий с одновременным сохранением обратной совместимости, удобочитаемости кода для человека и простоты анализа для парсеров [7].

*CSS* – формальный язык описания внешнего вида документа (веб-страницы), написанного с использованием языка разметки (чаще всего *HTML* или *XHTML*). Также может применяться к любым *XML*-документам, например, к *SVG* или *XUL*. *CSS* используется создателями веб-страниц для задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц [8].

*JavaScript* – мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации *ECMAScript* (стандарт *ECMA-262*) [9]. *JavaScript* обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

*Bootstrap* (также известен как *Twitter Bootstrap*) – свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя *HTML* и *CSS* шаблоны оформления для типографики, веб форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая *JavaScript* расширения. *Bootstrap* использует современные наработки в области *CSS* и *HTML*, поэтому необходимо быть внимательным при поддержке старых браузеров [10].

Проектируемая база данных логично задокументирована, так как название таблиц, атрибутов говорят об их назначении. Скрипт генерации базы данных представлен в приложении Б.

**3 ПРИМЕНЕНИЕ РАЗРАБОТАННОЙ БАЗЫ ДАННЫХ**

**3.1 Руководство пользователя**

**Руководство пользователя**

При первом входе на сайт пользователю будет показана страница розыска с кнопкой авторизации (см. рисунок 3.1).

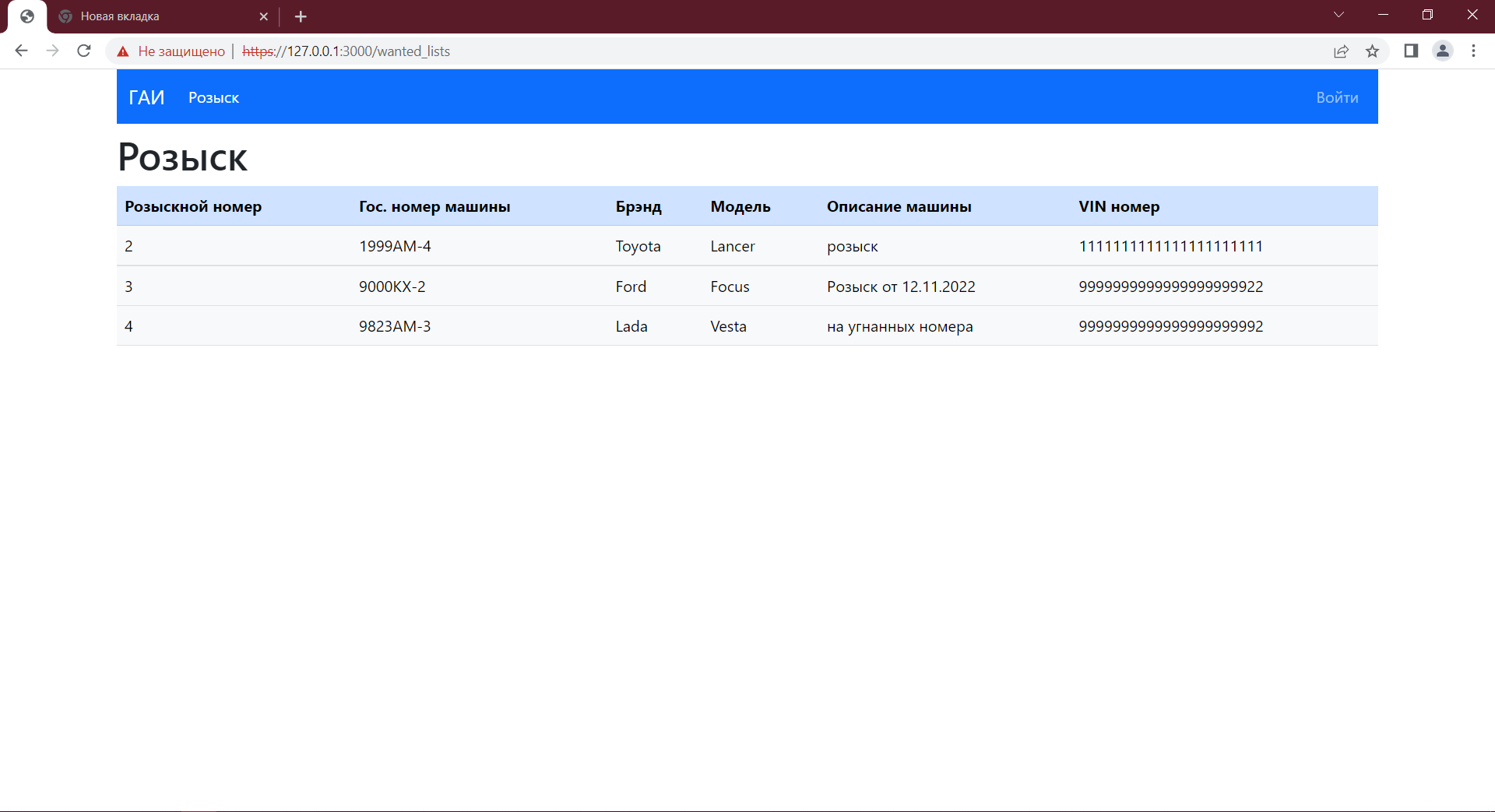
****

Рисунок 3.1 – Вывод главной страницы сайта ГАИ для неавторизованного пользователя

Так как существует разграничение прав пользователей, то и возможности у них различны. Неавторизованный пользователь может только просматривать базу «Розыск» и имеет возможность войти на сайт. Вот так выглядит страница входа, рисунок 3.2.

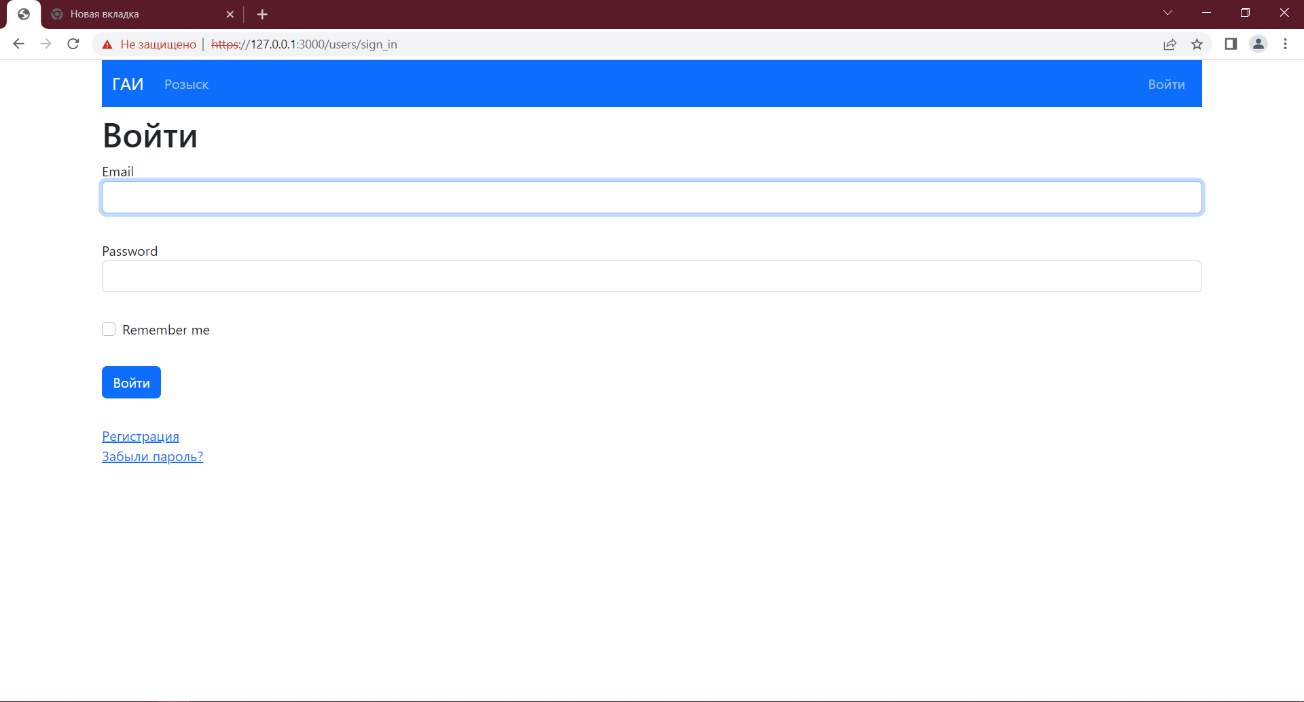


Рисунок 3.2 – Страница авторизации

Пользователь имеет возможность зарегистрироваться, используя электронную почту и пароль, рисунок 3.3.

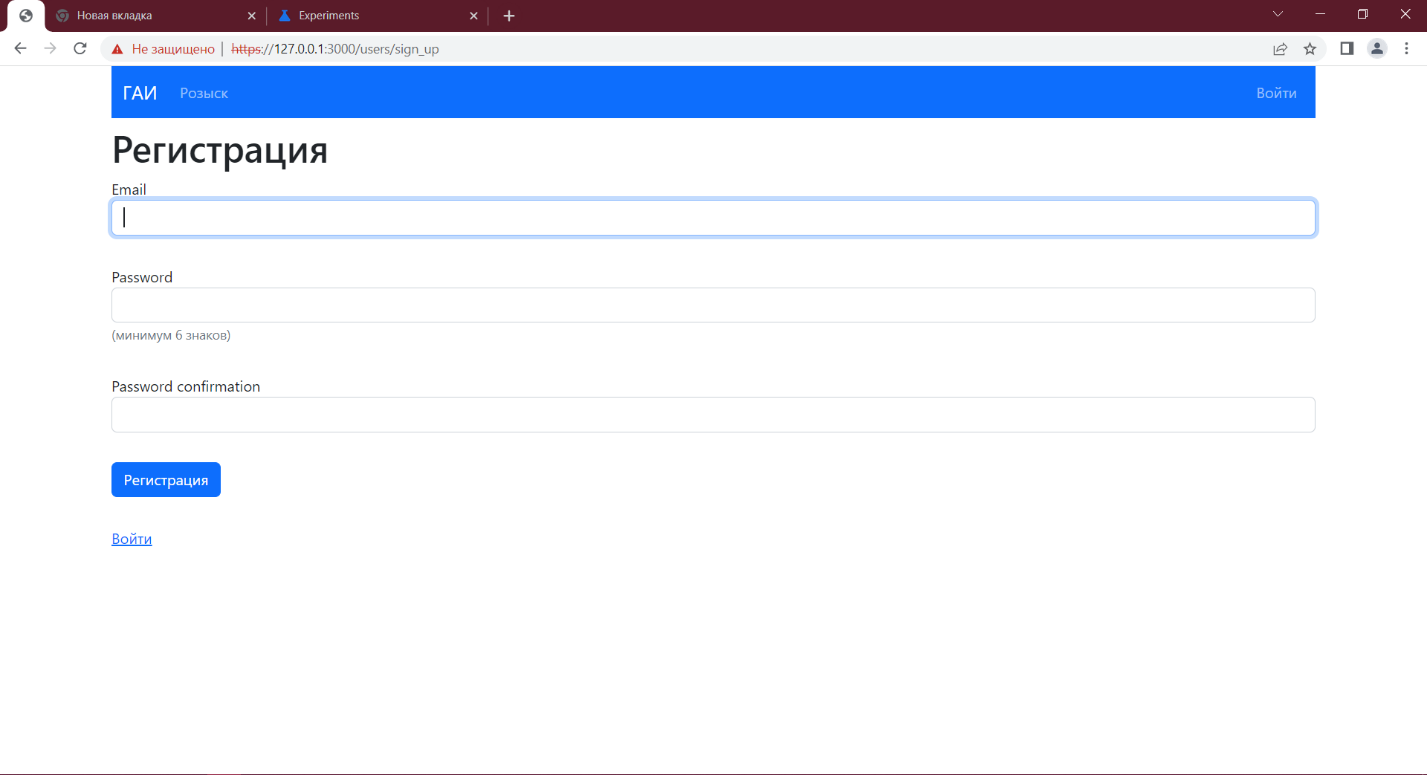


Рисунок 3.3 – Страница регистрации

Обычный зарегистрированный пользователь имеет все те же права, что и не зарегистрированный. Поменять права пользователя может либо начальник ГАИ, либо администратор базы данных. Пользователь не имеет прав добавлять, редактировать или удалять записи.

**Руководство работника**

Чтобы работник ГАИ смог редактировать записи в базах данных, ему необходимо авторизоваться на сайте, авторизация работника никак не отличается от авторизации пользователя, рисунок 3.1, 3.2.

После успешной авторизации работнику будут представлены дополнительный вкладки: «Информация об авто», «Владельцы авто», «Регистрация авто», «Штрафы на авто», «Жалобы на авто», рисунок 3.4.

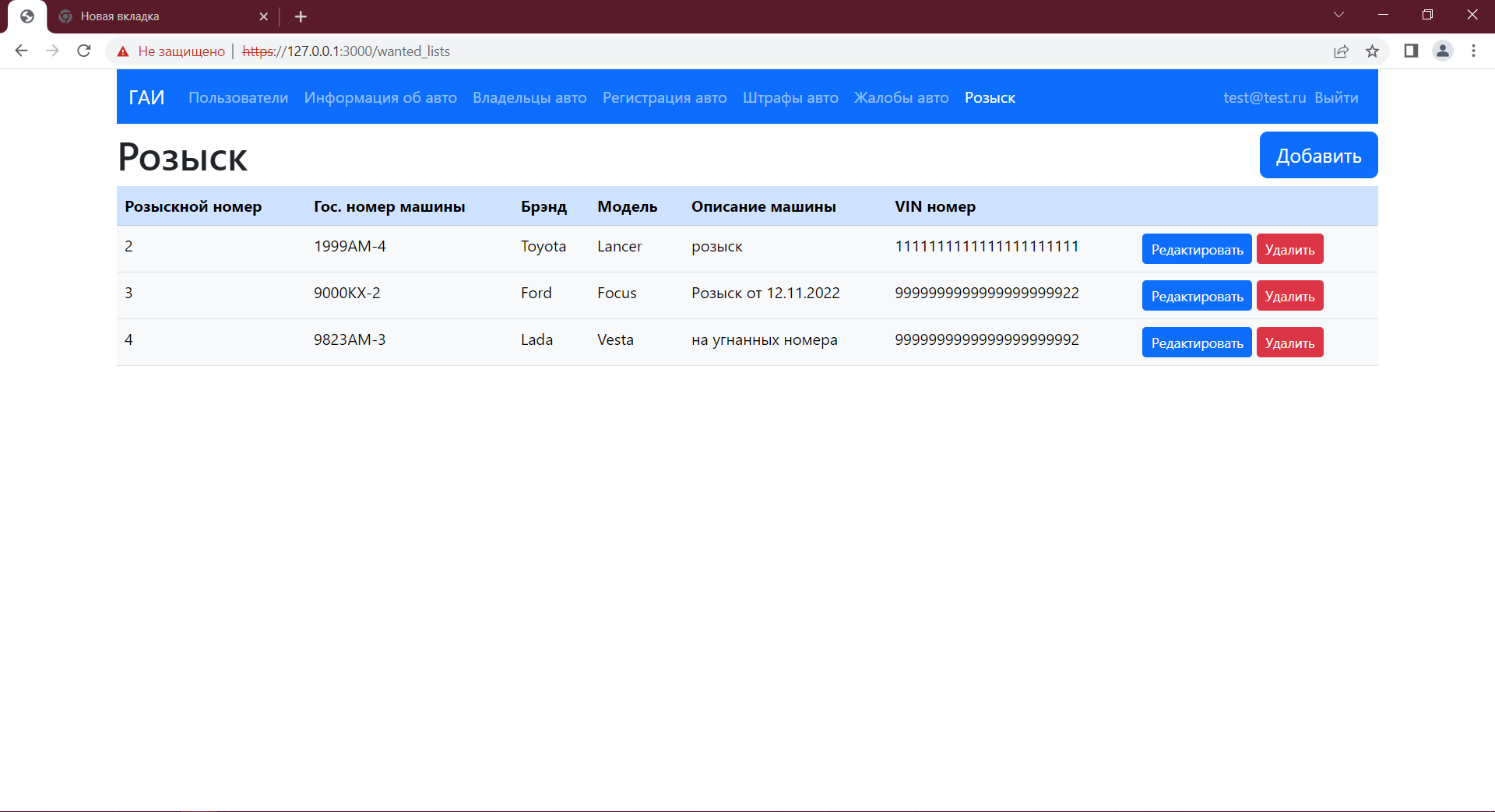


Рисунок 3.4 – Страница сайта после авторизации работником

Как видно на рисунке 3.4 работнику стали доступны дополнительные вкладки, а также кнопка «Редактировать» и «Добавить». Так будет выглядеть любая страница, например рисунок 3.5.

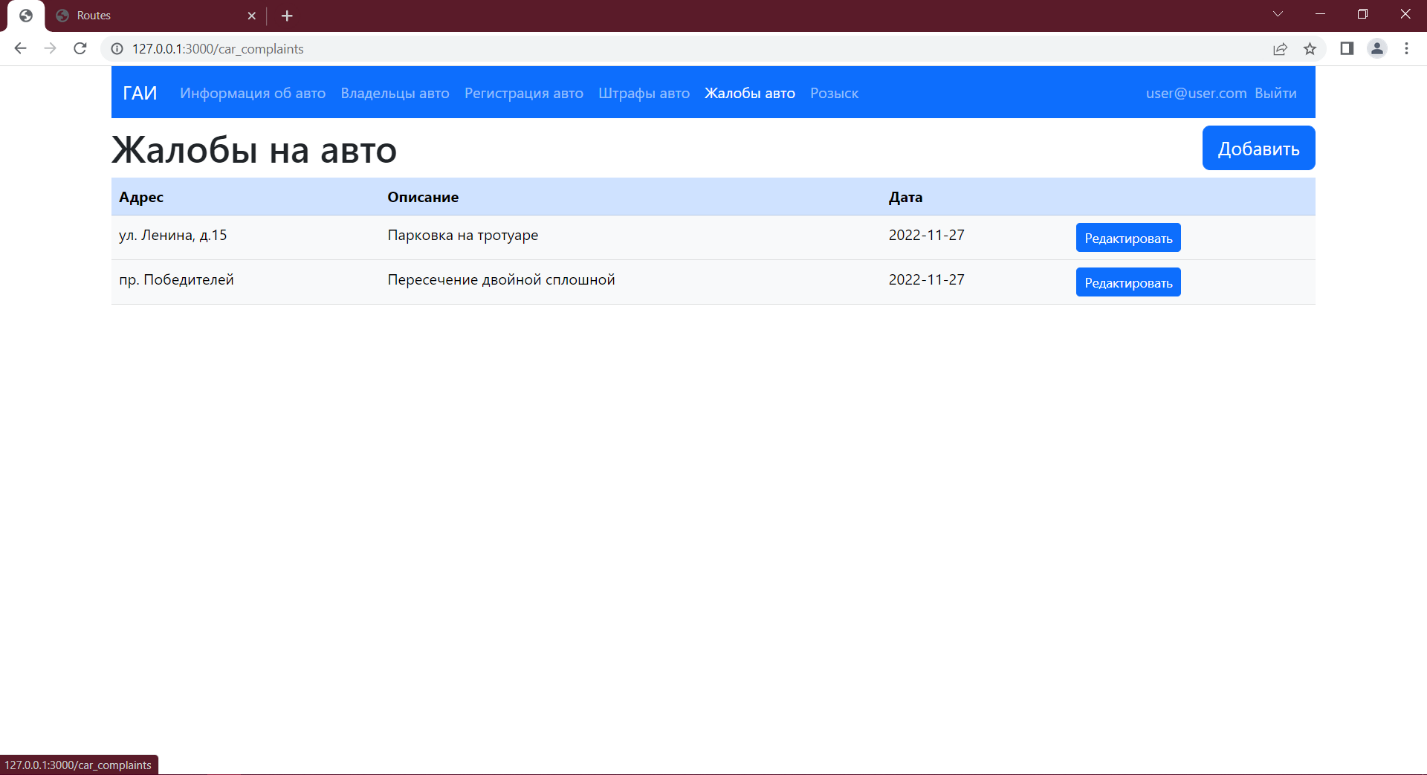


Рисунок 3.5 – Вид страницы «Жалобы авто» для сотрудника

Как упоминалось выше сотрудник может редактировать и добавлять записи на тех вкладках, которые ему доступны. Так выглядит страница добавления записи на вкладке «Розыск» рисунок 3.6.

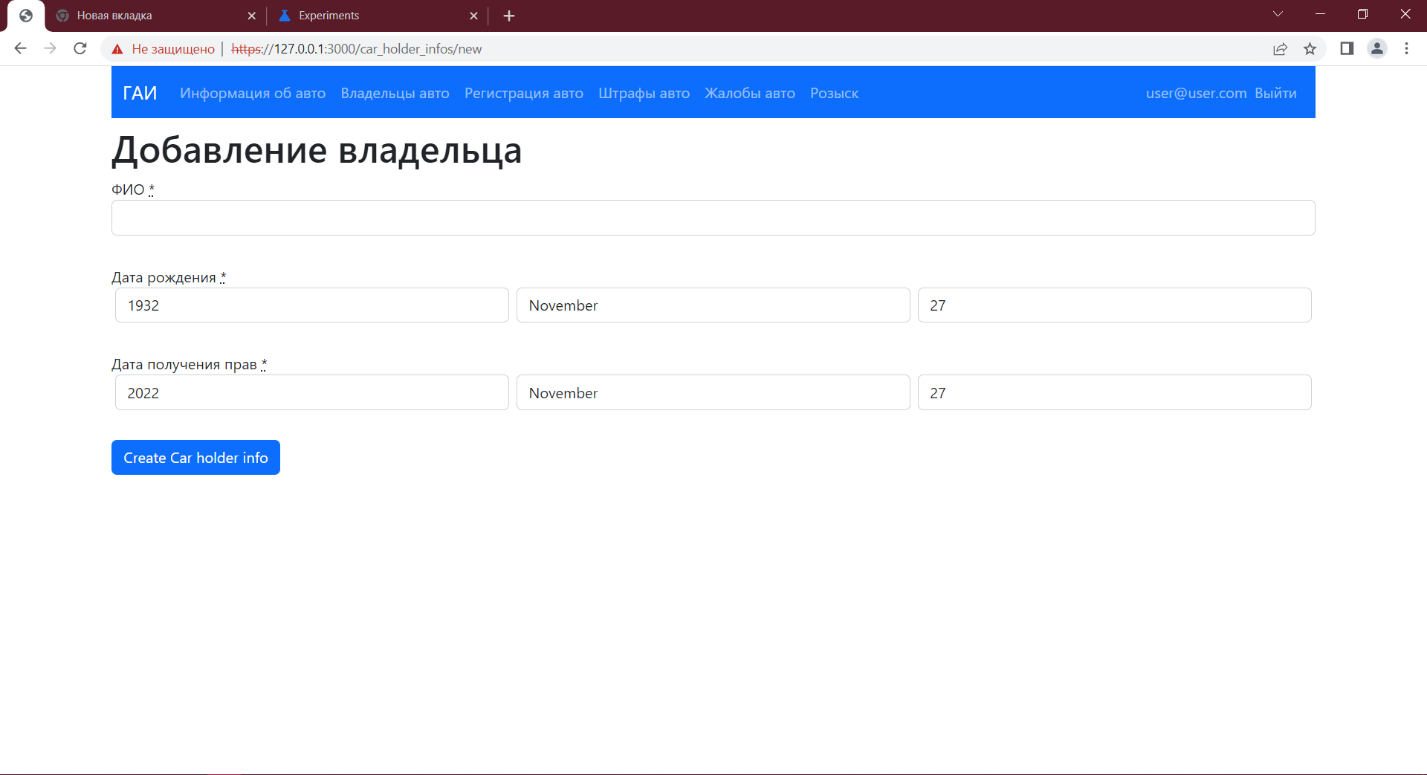


Рисунок 3.6 – Добавление записи для сотрудника

Также пользователь может редактировать уже существующие записи, для этого сделана кнопка «Редактировать», она находится рядом с каждой записью в таблице, как это выглядит можно наглядно посмотреть на рисунке 3.7.

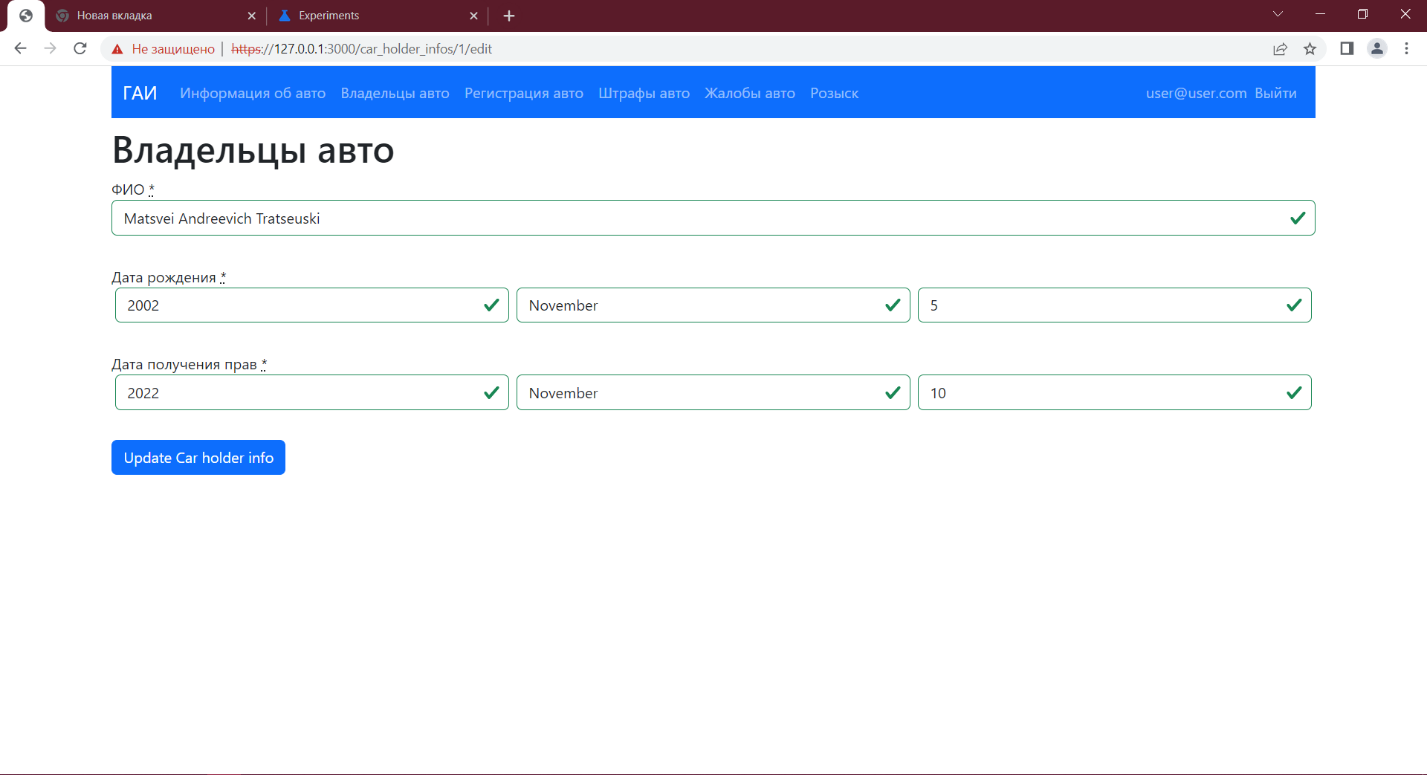


Рисунок 3.7 – Форма редактирования

Зелёные галочки показывают, что данные провалидированы и могут быть занесены в базу данных.

**3.2 Администрирование сайта**

Пользователями с наивысшим количеством прав являются начальник ГАИ и администратор базы данных. Различия. Он имеет доступ ко всем функция сотрудника, а также может изменять права во вкладке «Пользователи» всех пользователей и удалять записи в других таблицах. На рисунке 3.8 показана страница после авторизации начальником ГАИ. Разделение прав на уровне кода происходит в контроллере *users\_controller.rb*, код контроллера представлен в Приложении Б.

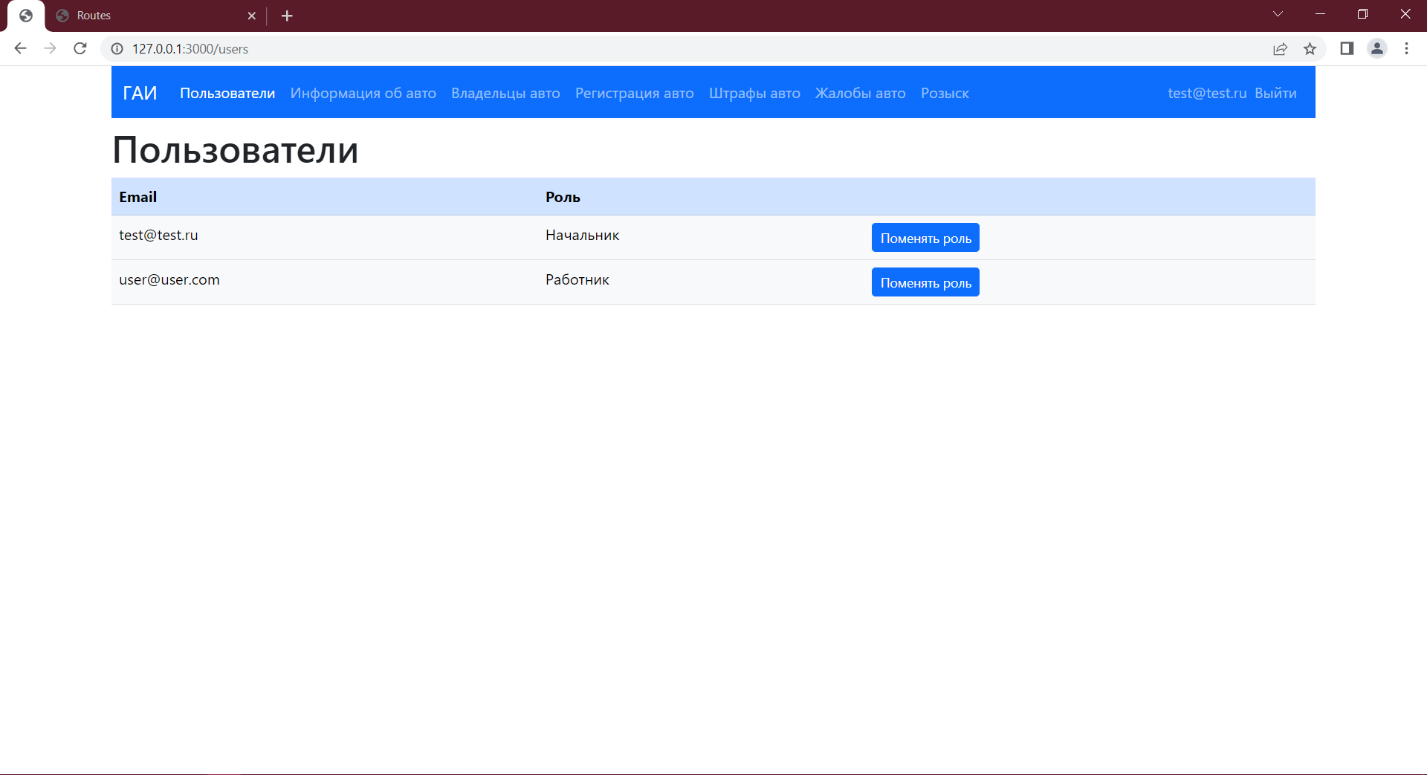


Рисунок 3.8 – Станица «Пользователи» доступная при авторизации начальником ГАИ

Поменять права пользователя начальник может в форме нажав на кнопку «Поменять роль» рисунок 3.9.

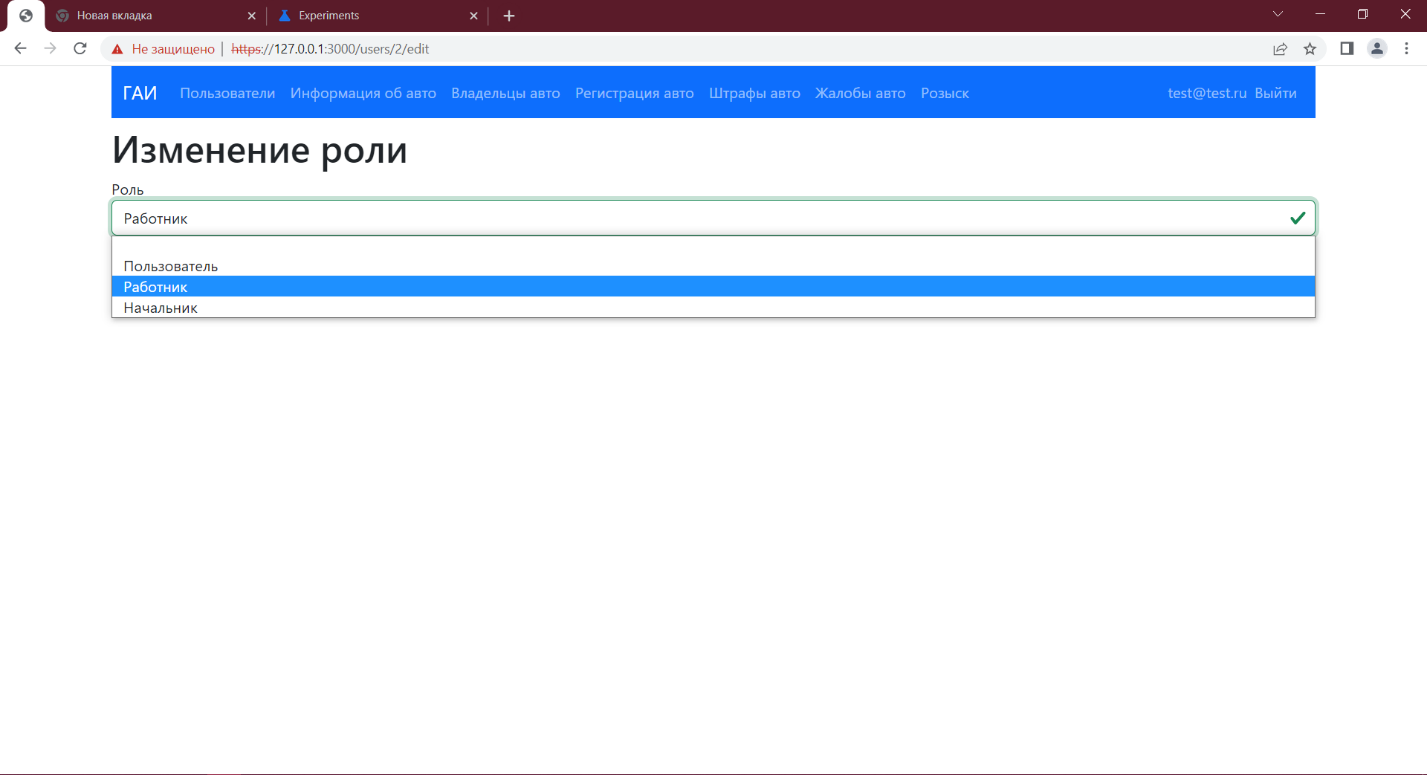


Рисунок 3.9 – Форма изменения роли

Как и упоминалось выше, у начальника есть возможность удаление записей в любой таблице, кнопка «Удалить» находится рядом с кнопкой «Изменить», это наглядно показано на рисунке 3.10.

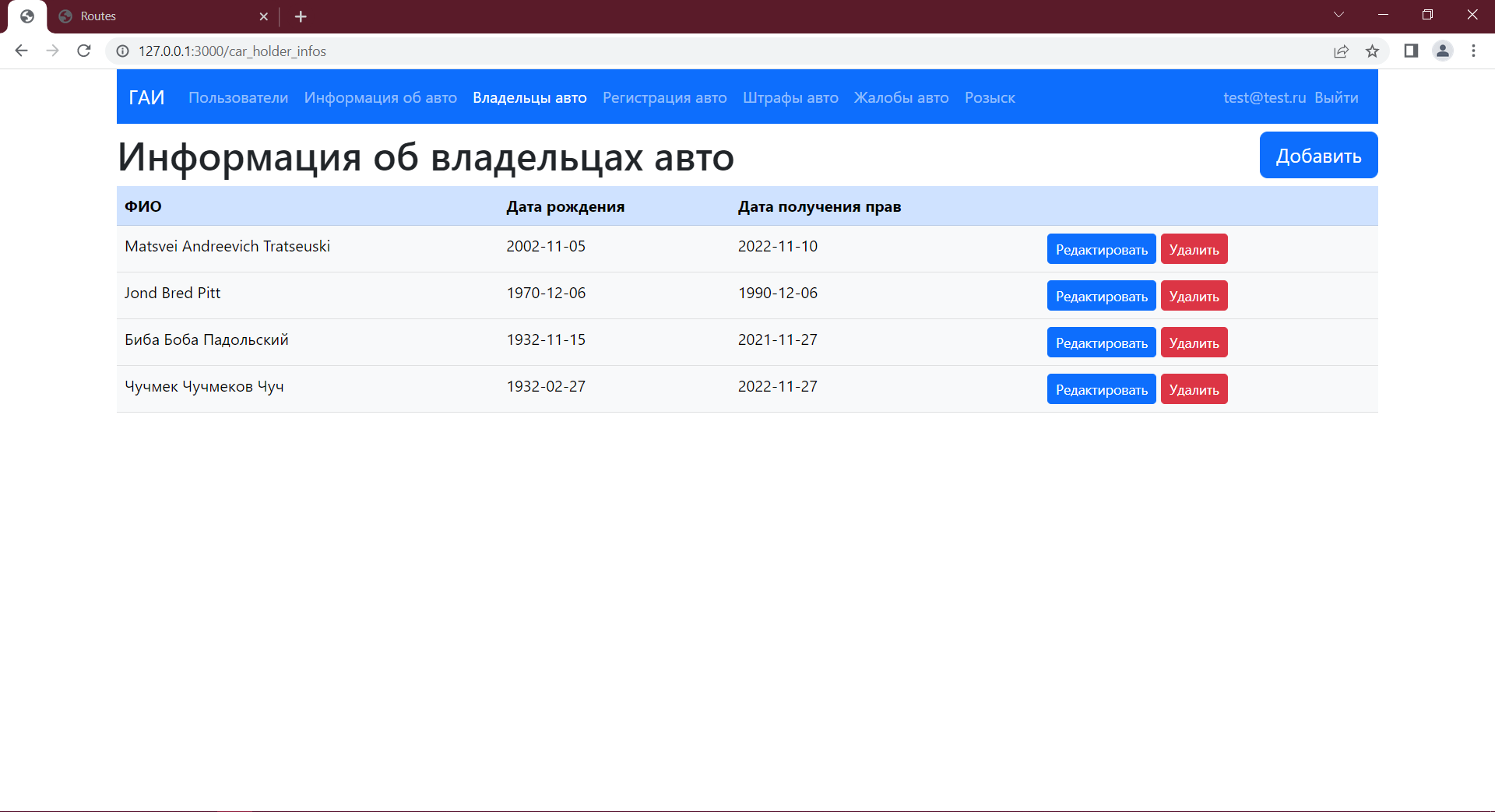


Рисунок 3.10 – Страница «Информация о владельцах авто» для начальника ГАИ

Администратор базы данных управляет ей с помощью консоли, в неё можно зайти с помощью команды *rails console* прописанной в командной строке в папке проекта рисунок 3.11.

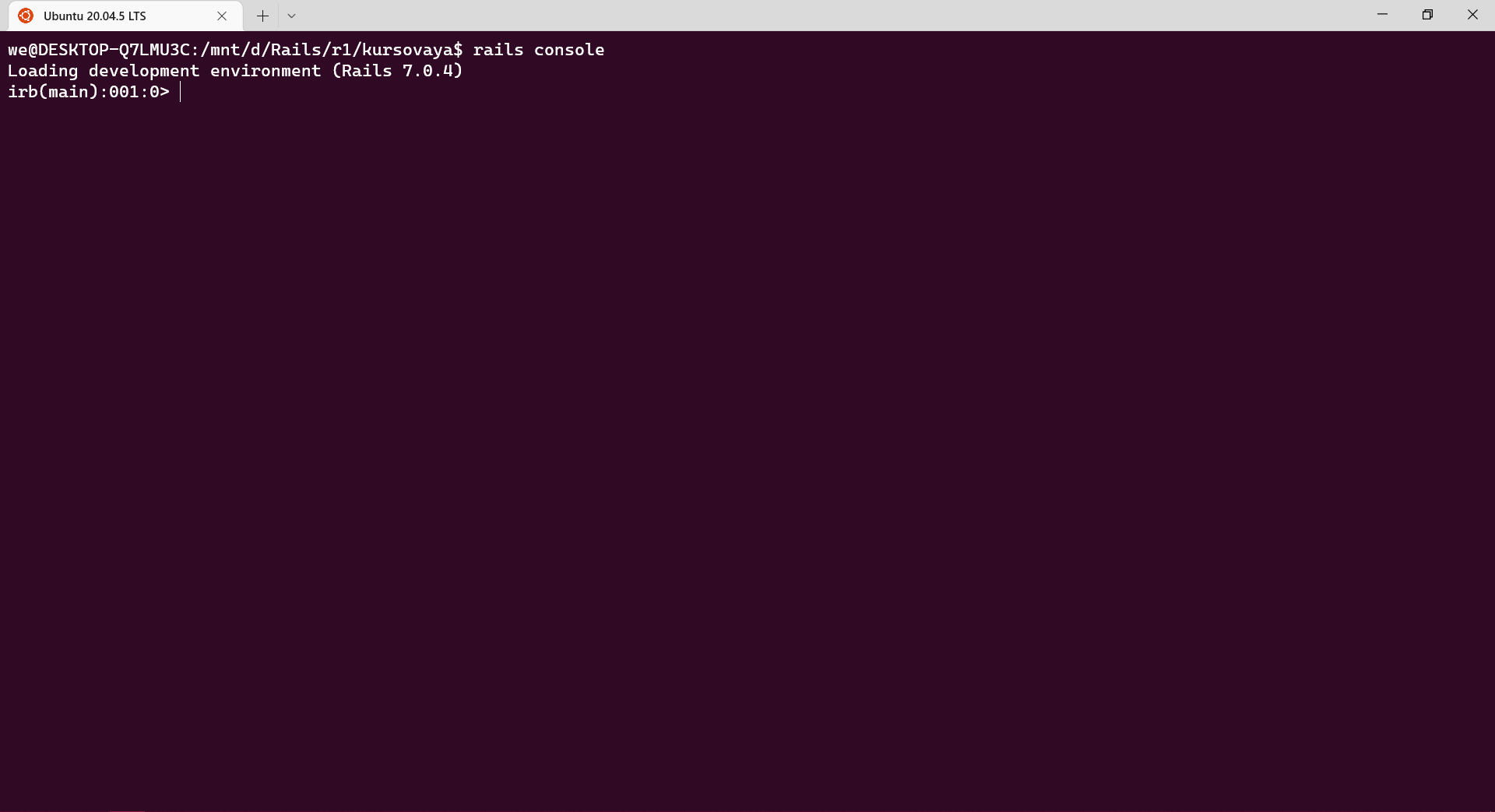


Рисунок 3.11 – Консоль *Rails* приложения

Из консоли можно полностью управлять базой данных. Команды и результат их выполнения предоставлен на рисунке 3.12.

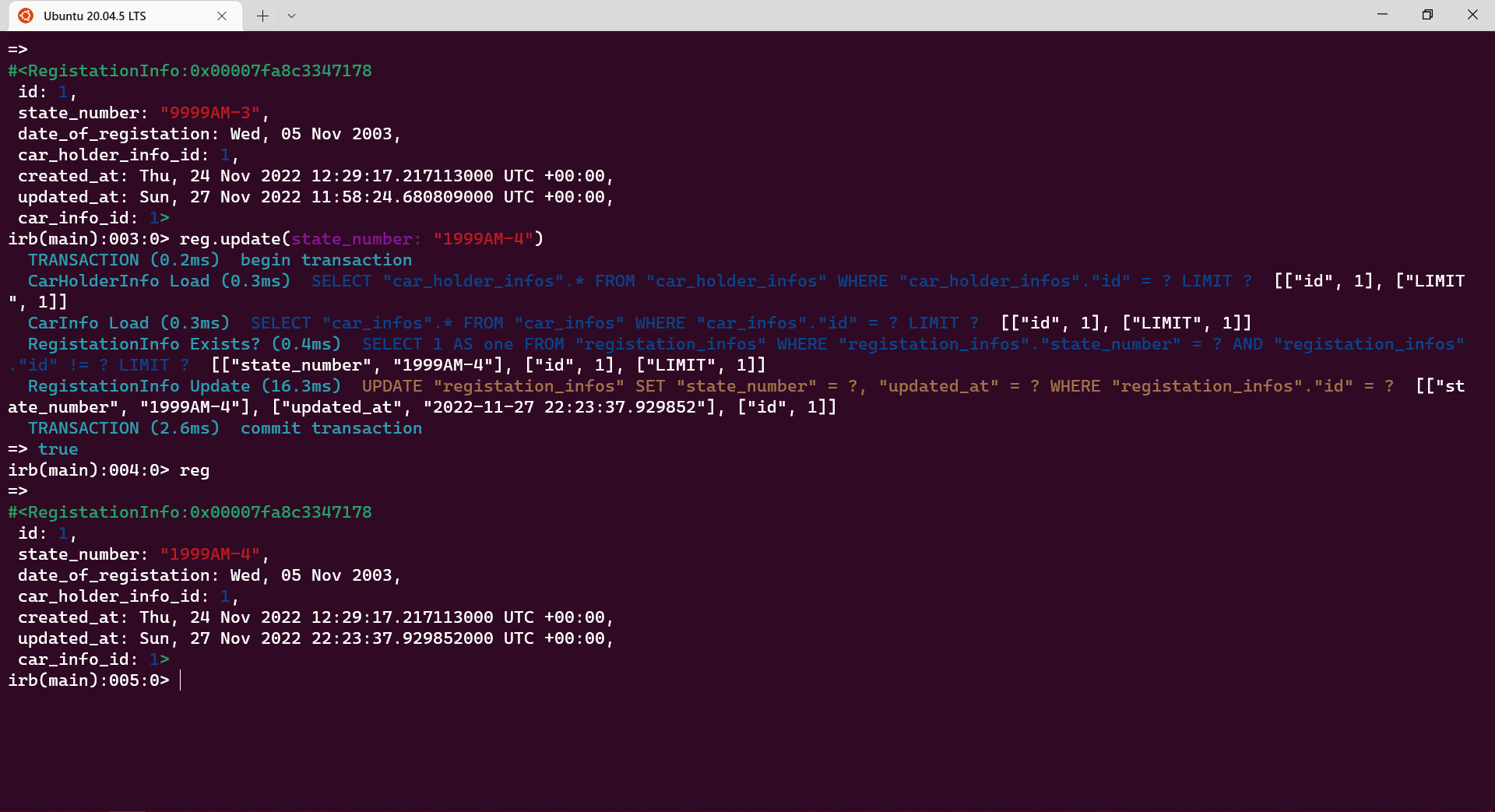


Рисунок 3.12 – Изменения номера машины с помощью консоли

На рисунке 3.12 показано, как можно поменять номер машины с помощью консоли.

**3.3 Реализация клиентских запросов**

Снова напомним, что база данных, как и приложение, делается в первую очередь для пользователя, который это приложение будет использовать. В данном разделе необходимо определить, какие пользователи будут чаще всего обращаться к одним данным, какие пользователи к другим данным и т.д. Для этого, прежде чем начинать делать приложение, следует разобраться в требовании заказчика и по мере получения этой информации продвигать идеи функционала приложения.

Приложение было разработано для организации работы ГАИ.

В приложении присутствуют все необходимые функции для работника ГАИ и начальника ГАИ, а именно просмотр, добавление, изменение и удаление из базы данных.

Работник имеет доступ к просмотру, редактированию и добавлению информации из базы данных. Начальник ГАИ имеет доступ к просмотру, редактированию, добавлению, удалению из базы данных, а также редактированию ролей доступа базы данных. Администратор базы данных имеет такой же уровень доступа, как и начальник, но для работы с базой данных использует консоль.

Обычный пользователь имеет доступ только к просмотру базы данных «Розыск».

**3.4 Обоснование и реализация механизма обеспечения** **безопасности и сохранности данных**

*HTTPS* – это безопасный протокол передачи данных, который поддерживает шифрование посредством криптографических протоколов *SSL* и *TLS*, и является расширенной версией протокола *HTTP*.

Сначала *HTTP* использовался только как протокол передачи гипертекста (текста с перекрёстными ссылками). Однако позже стало понятно, что он отлично подходит для передачи данных между пользователями. Протокол был доработан для новых задач и стал использоваться повсеместно.

Несмотря на свою функциональность у *HTTP* есть один очень важный недостаток – незащищённость. Данные между пользователями передаются в открытом виде, злоумышленник может вмешаться в передачу данных, перехватить их или изменить. Чтобы защитить данные пользователей, был создан протокол *HTTPS*.

*HTTPS* работает благодаря *SSL/TLS*-сертификату. *SSL/TLS*-сертификат – это цифровая подпись сайта. С её помощью подтверждается его подлинность. Перед тем как установить защищённое соединение, браузер запрашивает этот документ и обращается к центру сертификации, чтобы подтвердить легальность документа. Если он действителен, то браузер считает этот сайт безопасным и начинает обмен данными. Вот откуда взялась и что означает *S* в *HTTPS* [11].

Чтобы запустить *localhost* с использованием *HTTPS* надо сгенерировать ключи с расширениями *.srt* и *.key*, это было реализовано с помощью консольной утилиты *openssl*. Чтобы убедить, что *localhost* действительно использует *HTTPS* надо перейти в консоль браузера, открыть вкладку *SECURITY* рисунок 3.13.

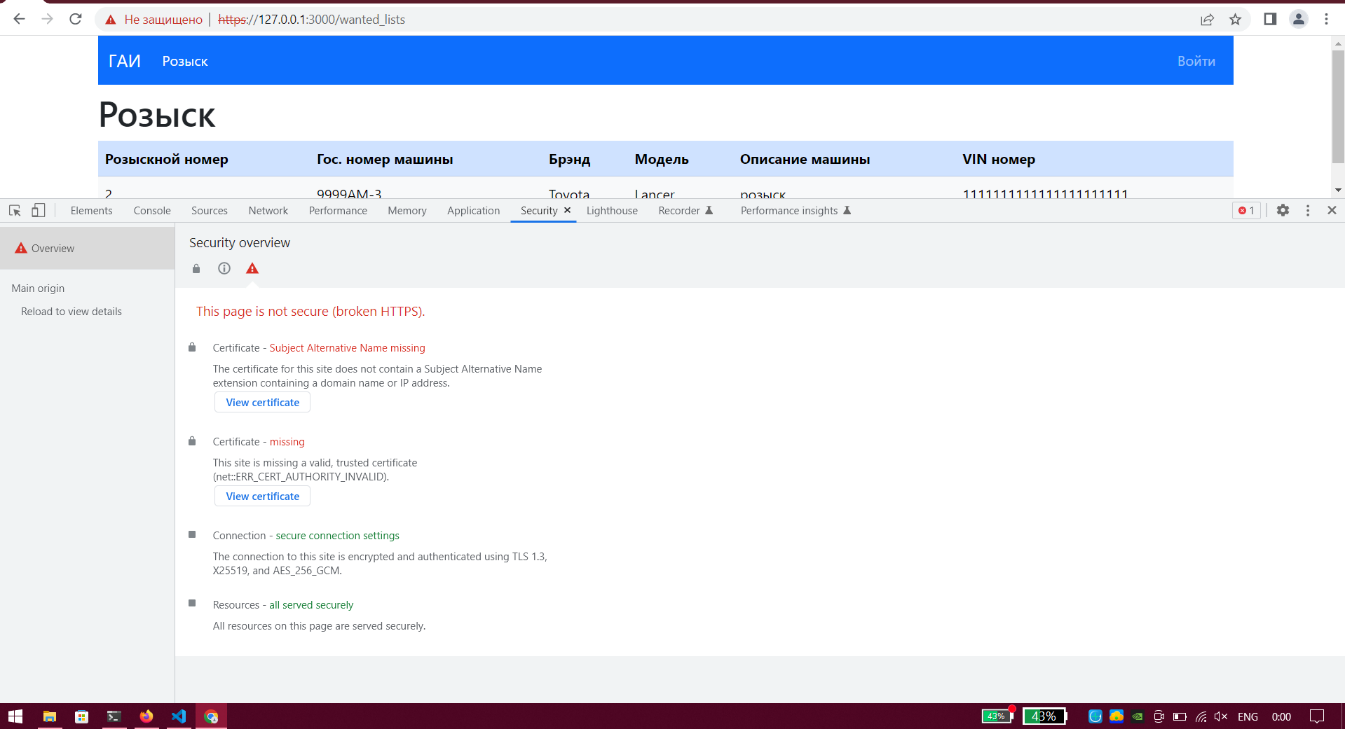


Рисунок 3.13 – Проверка наличия *HTTPS*

По умолчанию *Ruby on Rails* в режиме *development* запускает сервер с использованием *HTTP*. Это можно изменить, либо запустив сервер в режиме *production* либо поменять параметр *config.force\_ssl* в файле *development.rb*. В данной курсовой работе будет использован второй способ. Для того чтобы запустить сервер с использованием *HTTPS* в консоли надо прописать следующую команду: *rails server -b ‘\’\ssl://localhost:3000?key=mnt/d/Rails/r1/localhost.key&cert=/mnt/d/Rails/r1/locahost.crt’*. Для того чтобы не прописывать такую длинную команду каждый раз, был создан *alias* в файле *.bashrc.* Сам *alias*: *alias rss='rails server -b '\''ssl://localhost:3000?key=/mnt/d/Rails/r1/localhost.key&cert=/mnt/d/Rails/r1/localhost.crt'\'''*.

В данном случае браузер говорит, что подключение незащищено из-за того, что ключи были сгенерированы сами, а должны быть сгенерированы сторонней компанией. Тем не менее, весь трафик на сайте теперь будет шифрованным.

Разделение уровней доступа к базе данных сделано по средствам сторонней библиотеки *devise*, листинг кода контроллера представлен в Приложении Б.

**ЗАКЛЮЧЕНИЕ**

В данной курсовой работе была разработана автоматизированная информационная система для обеспечения работы Государственной автомобильной инспекции, которая полностью решает поставленную задачу и удовлетворяет всем поставленным требованиям.

На основании выполненных задач по курсу курсового проектирования можно сказать, что были исследованы объект (базы данных) и субъект (клиент серверные приложения).

Разработанная программа значительно облегчает работу, что выполняют сотрудники Государственной автомобильной инспекции, потому как автоматизирует её, и при этом экономит немало времени. При работе с гражданами востребовано – минимально сократить время на обслуживание каждого клиента и при этом максимально качественно обслужить его для чего и нужна данная программа.

Теоретическая деятельность реализовывалась на основе изучения информации по теме курсового проектирования, а также включающие в предметную область более глубокие темы, посредством поиска информации в лекционных материалах , учебно-методических пособиях и сети Интернет.

Разработанная база данных соответствует всем требованиям предметной области:

– требование полноты и непротиворечивости данных;

– многократное использование данных;

– быстрый поиск и получение информации по запросам пользователей;

– простота обновления данных;

– защита данных от несанкционированного доступа.

В процессе написания курсового проекта были получены и закреплены на практике навыки проектирования баз данных, приобретены умения построения клиент-серверного приложения на базе архитектуры *MVC*, а также навыки программирования сайтов, с понятным и удобным интерфейсом, и полной функциональностью.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Структуры и базы данных. Пособие для курсового проектирования [Электронный ресурс]. – Режим доступа: https://libeldoc.bsuir.by/bitstream/123456789/27723/1/Alekseev\_struk.pdf – Дата доступа: 8.10.2022

[2] Внешние ключи *FOREIGN KEY* [Электронный ресурс]. – Режим доступа: https://metanit.com/sql/mysql/2.5.php – Дата доступа 8.10.2022

[3] Нормализация баз данных [Электронный ресурс]. – Режим доступа: https://info-comp.ru/database-normalization – Дата доступа 8.10.2022

[4] Нормальные формы базы данных [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/254773/ – Дата доступа 8.10.2022

[5] *Ruby on Rails Active Record* что такое и с чем его едят [Электронный ресурс]. – Режим доступа: http://rusrails.ru/active-record-basics/ – Дата доступа 8.10.2022

[6] *Ruby on Rails — A web-app framework that includes everything needed to create database-backed web applications according to the Model-View-Controller (MVC) pattern* [Электронный ресурс]. – Режим доступа: https://rubyonrails.org/ – Дата доступа 8.10.2022

[7] *W3C HTML* [Электронный ресурс]. – Режим доступа: https://www.w3.org/html/ – Дата доступа 8.10.2022

[8] [Электронный ресурс]. – Режим доступа: https://www.w3.org/Style/CSS/Overview.en.html – Дата доступа 8.10.2022

[9] *Cascading Style Sheets* [Электронный ресурс]. – Режим доступа: – Дата доступа 8.10.2022

[10] *Bootstrap The most popular HTML, CSS, and JS library in the world* [Электронный ресурс]. – Режим доступа: https://getbootstrap.com/ – Дата доступа 8.10.2022

[11] Что такое протокол HTTPS и принципы его работы [Электронный ресурс]. – Режим доступа: https://help.reg.ru/support/ssl-sertifikaty/obshchaya-informatsiya-po-ssl/chto-takoye-protokol-https-i-printsipy-yego-raboty – Дата доступа 8.10.2022

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Скрипт генерации БД**

*ActiveRecord::Schema[7.0].define(version: 2022\_11\_26\_222051) do*

*create\_table "car\_complaints", force: :cascade do |t|*

*t.string "address"*

*t.text "description"*

*t.date "date"*

*t.integer "registation\_info\_id", null: false*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*t.index ["registation\_info\_id"], name: "index\_car\_complaints\_on\_registation\_info\_id"*

*end*

*create\_table "car\_holder\_infos", force: :cascade do |t|*

*t.string "full\_name"*

*t.date "date\_of\_birth"*

*t.date "date\_of\_car\_license"*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*end*

*create\_table "car\_infos", force: :cascade do |t|*

*t.string "VIN"*

*t.string "brand"*

*t.string "model"*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*end*

*create\_table "fines", force: :cascade do |t|*

*t.date "date\_of\_recepeit"*

*t.string "article"*

*t.integer "amount"*

*t.date "date\_of\_payment"*

*t.integer "registation\_info\_id", null: false*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*t.index ["registation\_info\_id"], name: "index\_fines\_on\_registation\_info\_id"*

*end*

*create\_table "registation\_infos", force: :cascade do |t|*

*t.string "state\_number"*

*t.date "date\_of\_registation"*

*t.integer "car\_holder\_info\_id", null: false*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*t.integer "car\_info\_id", null: false*

*t.index ["car\_holder\_info\_id"], name: "index\_registation\_infos\_on\_car\_holder\_info\_id"*

*t.index ["car\_info\_id"], name: "index\_registation\_infos\_on\_car\_info\_id"*

*end*

*create\_table "users", force: :cascade do |t|*

*t.string "email", default: "", null: false*

*t.string "encrypted\_password", default: "", null: false*

*t.string "reset\_password\_token"*

*t.datetime "reset\_password\_sent\_at"*

*t.datetime "remember\_created\_at"*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*t.string "role"*

*t.index ["email"], name: "index\_users\_on\_email", unique: true*

*t.index ["reset\_password\_token"], name: "index\_users\_on\_reset\_password\_token", unique: true*

*end*

*create\_table "wanted\_lists", force: :cascade do |t|*

*t.text "description"*

*t.integer "car\_info\_id", null: false*

*t.datetime "created\_at", null: false*

*t.datetime "updated\_at", null: false*

*t.integer "registation\_info\_id"*

*t.index ["car\_info\_id"], name: "index\_wanted\_lists\_on\_car\_info\_id"*

*t.index ["registation\_info\_id"], name: "index\_wanted\_lists\_on\_registation\_info\_id"*

*end*

*add\_foreign\_key "car\_complaints", "registation\_infos"*

*add\_foreign\_key "fines", "registation\_infos"*

*add\_foreign\_key "registation\_infos", "car\_holder\_infos"*

*add\_foreign\_key "registation\_infos", "car\_infos"*

*add\_foreign\_key "wanted\_lists", "car\_infos"*

*add\_foreign\_key "wanted\_lists", "registation\_infos"*

*end*

# ПРИЛОЖЕНИЕ Б

**(обязательное)**

**Листинги программного кода**

*class UsersController < ApplicationController*

*before\_action :find\_user, only: %i(edit update)*

*before\_action :authorize\_user*

*def index*

*@users = User.all*

*end*

*def edit*

*end*

*def update*

*if @user.update(role\_params)*

*redirect\_to action: :index*

*else*

*render 'edit'*

*end*

*end*

*private*

*def authorize\_user*

*authorize!(@user || User.new)*

*end*

*def find\_user*

*@user = User.find(params[:id])*

*end*

*def role\_params*

*params.require(:user).permit(:role)*

*end*

*end*

class CarInfosController < ApplicationController

before\_action :find\_car, except: [:index, :new, :create]

before\_action :authorize\_car

def index

@cars = CarInfo.all

end

def new

@car = CarInfo.new

end

def create

@car = CarInfo.new(car\_params)

if @car.valid?

@car.save

redirect\_to action: :index

else

render 'new'

end

end

def edit

end

def update

@car.assign\_attributes(car\_params)

if @car.valid?

@car.save

redirect\_to action: :index

else

render 'edit'

end

end

def destroy

@car.destroy

redirect\_to action: :index

end

def registation\_infos

registation\_infos = @car.registation\_infos.map do |registation\_info|

{

id: registation\_info.id,

state\_number: registation\_info.state\_number

}

end

render json: registation\_infos, status: :ok

end

private

def authorize\_car

authorize!(@car || CarInfo.new)

end

def find\_car

@car = CarInfo.find(params[:id])

end

def car\_params

params.require(:car\_info).permit(:VIN, :brand, :model)

end

end

class ApplicationController < ActionController::Base

before\_action :authenticate\_user!

end

# ПРИЛОЖЕНИЕ В

# (обязательное)

**Отчёт о проверке на заимствования в системе «Антиплагиат»**



Рисунок В.1 – Проверка в системе «Антиплагиат»

# ПРИЛОЖЕНИЕ Г

**(обязательное)**

**Ведомость курсового проекта**