Министерство образования и науки Республики Татарстан

Государственное автономное профессиональное образовательное учреждение

«Технический колледж им. В.Д. Поташова»

09.02.07 Информационные системы и программирование

МДК 01.01 Разработка программных модулей

КУРСОВАЯ РАБОТА

На тему: Разработка программного модуля для салона автомобилей\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил студент \_\_IV\_\_ курса, \_\_\_\_\_Денисов Тимур Юрьевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ф.И.О.

Шифр \_РК.5572.9-2.10.ПЗ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель преподаватель профессионального цикла, Миннуллина Р.Г.\_\_\_\_\_\_\_\_\_\_\_\_

должность, звание Ф.И.О.\_

К защите Защита принята с оценкой

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2022\_\_г. \_\_\_\_\_«\_\_\_\_»\_\_\_\_\_\_ 2022\_\_г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись Подпись

Набережные Челны, 2022 год

СОДЕРЖАНИЕ

[Введение 3](#_Toc100317209)

[1 Описание предметной области 5](#_Toc100317210)

[1.1 Исследование предметной области 5](#_Toc100317211)

[1.2 Описание структуры базы данных 6](#_Toc100317212)

[1.3 Обоснование выбора языка программирования 8](#_Toc100317213)

[2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 10](#_Toc100317214)

[2.1 Описания интерфейса приложения 10](#_Toc100317215)

[2.2 Инструкция для пользователя 10](#_Toc100317216)

[2.3 Тестирование и отладка приложения 15](#_Toc100317217)

[Заключение 16](#_Toc100317218)

[Список используемой литературы 17](#_Toc100317219)

[ПРИЛОЖЕНИЯ 18](#_Toc100317220)

# Введение

Актуальность состоит в том, что у любой современной компании существует сайт. Это один из способов привлечения клиентов ведь именно в интернете потенциальные клиенты будут искать информацию о компании. Также сайт поможет упростить обмен информации между менеджерами и клиентами.

Практическая значимость обусловлена возможностью систематизировать и упорядочить информацию о представленных автомобилях и услугах, а также обмениваться информацией между клиентом и сотрудником посредством вывода информации об автомобиле в удобным виде для восприятия.

Целью курсовой работы является:

* систематизации и закрепления полученных теоретических знаний и практических умений по профессиональному модулю ПМ.01 Разработка программных модулей программного обеспечения для компьютерных систем;
* формирования умений применять теоретические знания при решении поставленных вопросов;
* приобретение практических навыков по написанию программ на языке объектно-ориентированного программирования;
* овладение профессиональными (ПК) компетенциями:
* ПК 1.1 Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
* ПК 1.2 Разрабатывать программные модули в соответствии с техническим заданием;
* ПК 1.3 Выполнять отладку программных модулей с использованием специализированных программных средств;

Задачами курсовой работы являются:

* рассмотреть теоретические аспекты разработки веб-сайта, относящихся к теме курсовой работы;
* разработать веб-сайт;
* анализ возможных подходов и методов решения с обоснованием выбранного метода;
* выбор или разработка информационной модели, необходимой для достижения цели;
* разработать интуитивный и удобный пользовательский интерфейс;
* разработка модуля информационной системы, включающая как проектирование баз данных, так и создание программного модуля в виде веб-сайта;
* разработка программной и эксплуатационной документации.

В итоге главной задачей курсовой работы является создание надежной, эффективной и незаменимой в работе системы, которая будет в помощь сотрудникам и клиентам салона автомобилей по предоставлению автомобилей и услуг.

# 1 Описание предметной области

## Исследование предметной области

На сегодняшний день автомобильный рынок в развивается стабильно. С каждым годом количество автосалонов увеличивается. Это объясняется ростом спроса на автомобили, а также невысокими барьерами при входе на рынок. Вместе с увеличением числа автосалонов наблюдается и рост уровня конкуренции.

Результатом увеличивающегося числа автосалонов является еще и то, что барьеры при входе на рынок не слишком высоки. Единственным препятствием для осуществления деятельности являются большие затраты на покупку автомобилей, закупку площадей, регистрацию фирмы и осуществление маркетинговой деятельности.

Таким образом, конкуренция на авторынке города очень высокая, что объясняется борьбой автосалонов за один сегмент потребителей, невысокими барьерами при входе на рынок, а также возрастающим спросом на автомобили.

Основным видом деятельности автосалона является реализация автомобилей.

Автосалон - юридическое лицо, имеет зарегистрированный товарный знак, фирменный знак, расчетный и иные счета в рублях и иностранной валюте в учреждениях банков.

Автоматизированная система для менеджера автосалона предназначена для автоматизации деятельности автомобильного салона. Разрабатываемая система должна обеспечивать решение следующих задач:

автоматизировать процесс оформления продажи автомобилей;

повысить точность, сохранность и защищённость данных;

[ускорить процесс поиска автомобилей](https://topuch.ru/informacionnoe-obespechenie-kontraktnoj-sistemi/index.html) по маркам и моделям;

повысить удобство работы с базой данных;

осуществить контроль вводимых данных;

систематизировать хранение данных о клиентах, автомобилях;

упростить процедуру ввода данных.

Дизайн должен быть выполнен в спокойных тонах не напрягающий глаза пользователей. Расположение кнопок на форме – максимально удобной для работы. При разработке дизайна необходимо учитывать ряд общепринятых правил: автоматизированная система сочетание цветов, пропорциональность размеров элементов, интуитивно понятный интерфейс.

Разрабатываемый сайт имеет следующую функциональную схему (Рисунок 1).

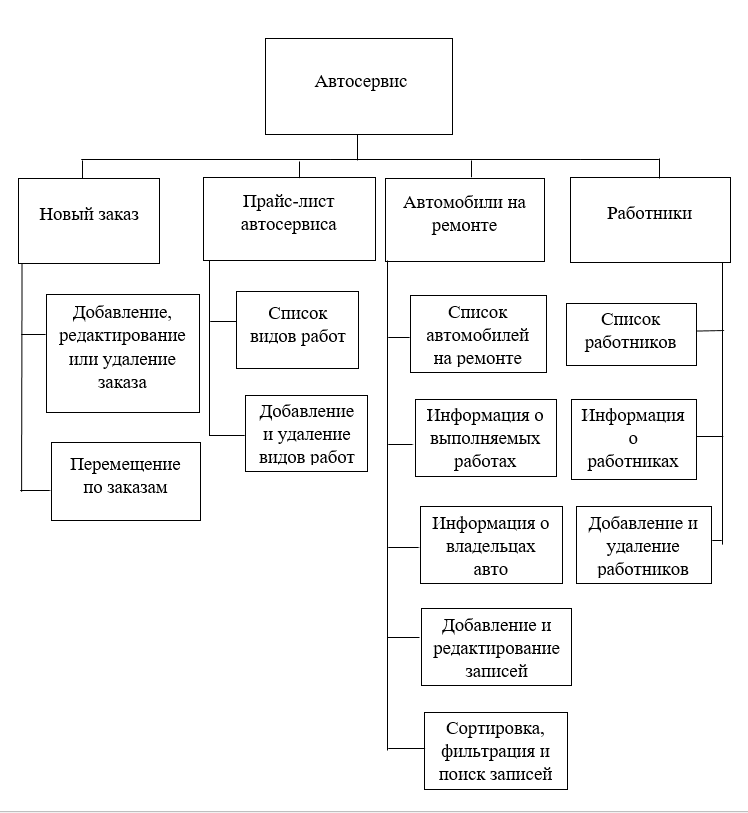


Рисунок 1 – Функциональная схема веб-сайта

## Описание структуры базы данных

После исследования предметной области и разработки постановки задачи, для хранения вводимых данных были созданы следующие таблицы:

* «users» - содержит информацию о менджерах;
* «cars» - содержит данные об автомобилях;
* «brands» - содержит данные о марках моделях;
* «avto\_m\_s» - содержит данные о моделях;
* «clients» - содержит данные о клиентах.

Таблица 1 Описание полей таблицы «users»

|  |  |  |
| --- | --- | --- |
| Наименование поля | Тип | Длина |
| id | Счетчик | 10 |
| name | Текстовый | 50 |
| login | Текстовый | 60 |
| email | Текстовый | 15 |
| password | Текстовый | 20 |
| is\_admin | Числовой | 10 |

Таблица 2 Описание полей таблицы «cars»

|  |  |  |
| --- | --- | --- |
| Наименование поля | Тип | Длина |
| id | Счетчик | 10 |
| brand\_id | Текстовый | 25 |
| model | Текстовый | 20 |
| year | Дата | Дата |
| probeg | Текстовый | 30 |
| xoz | Текстовый | 10 |
| opis | Текстовый | 30 |
| price | Числовой | 20 |

Таблица 3 Описание полей таблицы «brands»

|  |  |  |
| --- | --- | --- |
| Наименование поля | Тип | Длина |
| id | Счетчик | 10 |
| addbrand | Текстовый | 50 |

Таблица 4 Описание полей таблицы «avto\_m\_s»

|  |  |  |
| --- | --- | --- |
| Наименование поля | Тип | Длина |
| id | Числовой | 30 |
| brand\_id | Числовой | 50 |
| models | Текстовой | 50 |

Таблица 5 Описание полей таблицы «clients»

|  |  |  |
| --- | --- | --- |
| Наименование поля | Тип | Длина |
| id | Числовой | 30 |
| fio\_client | Текстовый | 50 |
| number\_client | Числовой | 15 |
| status\_client | Текстовый | 50 |

База данных которой представлена в виде MySQLотображена на Рисунок 2.

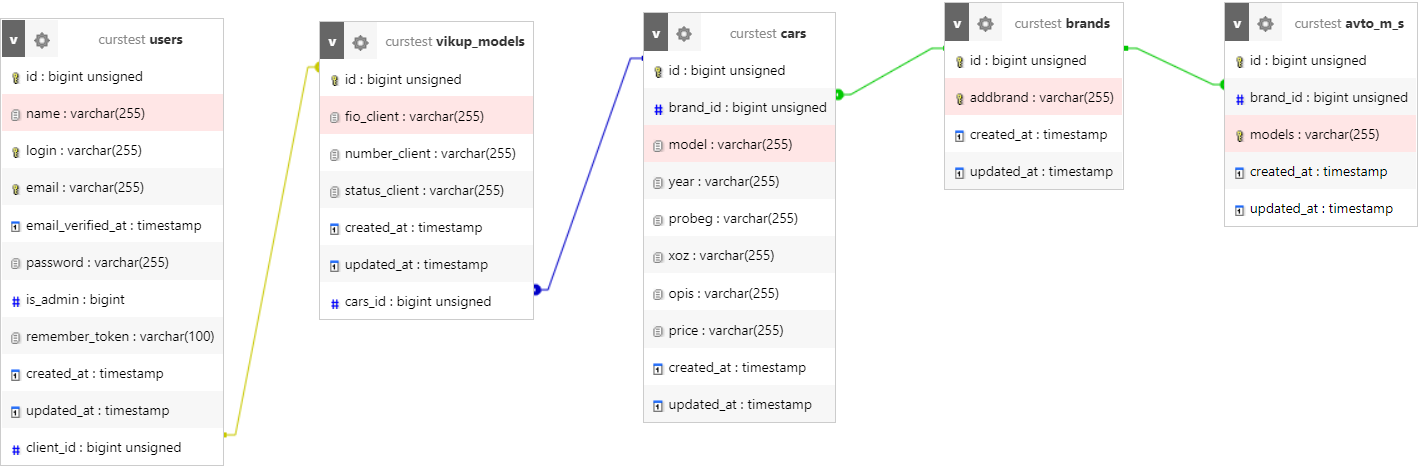


Рисунок 2 **–** Схема базы данных сайта

## 1.3 Обоснование выбора языка программирования

Для работы интернет-магазина необходим web-сервер, который можно установить практически на любую операционную систему: Windows.В текущем проекты будет использоваться операицонная система Windows10.

В качестве локального сервера использовался OpenServer.

В качестве системы управления базами данных (СУБД) выбрана MySQL под Windows.

Кроме вышеперечисленного программного обеспечения для работы с web-приложением компьютеры пользователей (клиентов) должны быть оснащены программами для просмотра web-сайтов, например:

* Google Chrome. Браузер, разрабатываемый компанией Googleна основе свободного браузера Chromium и движка Blink;
* Mozilla Firefox. Браузер нового поколения от Mozilla Foundation, обладающий высокой степенью безопасности, скорости работы, гибкостью и расширяемостью;
* Opera. Web-браузер и пакет прикладных программ для работы в Интернете, выпускаемый компанией Opera Software.

Laravel («Ларавел») – это бесплатный PHP-фреймворк с открытым исходным кодом, специально разработанный для создания сложных сайтов и веб-приложений. Позволяет упростить аутентификацию, маршрутизацию, сессии, кэширование, архитектуру приложения, работу с базой данных.

Laravel пользуются [backend-разработчики](https://skillfactory.ru/backend-razrabotchik-na-golang?utm_source=blog&utm_medium=referral&utm_campaign=Laravel&utm_content=coding_go&utm_term=text), которые пишут код на [PHP](https://blog.skillfactory.ru/glossary/php/). Он помогает определить структуру веб-приложения и служит для нее каркасом. Фреймворк написан на PHP и расширяет его возможности. Распространяется бесплатно под лицензией MIT.

Назначение Laravel – создание веб-приложений и сайтов на основе MVC (model-view-controller). Это вариант архитектуры, при котором компоненты программы делятся на три части:

* модель (model) предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность;
* представление (view) показывает пользователю эти данные и изменяется, если меняется модель;
* контроллер (controller) направляет данные от пользователя к системе и наоборот.

Когда пользователь работает с приложением, построенным по схеме MVC, он взаимодействует с представлением и контроллером. Представление - это то, что он видит, например сведения, которые отображены в визуальном интерфейсе. А контроллеру пользователь отдает команды.

Архитектура MVC позволяет писать более читабельный код, а процесс разработки сделать комфортным, разграничивая работу frontend- и backend-разработчиков.

Возможности фреймворка Laravel:

* консоль Artisan;
* Eloquent ORM;
* конструктор Fluent;
* шаблонизатор Blade;
* валидация;
* система контроля версий баз данных;
* аутентификация.

Особенности Laravel:

* высокая производительность;
* безопасность;
* открытый код;
* понятный синтаксис;
* гибкая маршрутизация;
* лёгкая миграция;
* обилие пакетов и библиотек.

# 2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 2.1 Описания интерфейса приложения

Веб-интерфейс — это совокупность средств, при помощи которых пользователь взаимодействует с веб-сайтом или любым другим приложением через браузер. Веб-интерфейсы получили широкое распространение в связи с ростом популярности всемирной паутины и соответственно — повсеместного распространения веб-браузеров.

Веб-интерфейс применяют для работы с различными онлайн-сервисами. Это может быть обычная электронная почта или специальные системы веб-аналитики. Иногда на сайтах веб-интерфейс именуют «Личным кабинетом».

Само понятие говорит за себя. Так, приставка «веб» означает удаленную работу, вдали от компьютера пользователя, на интернет-сервере. Взаимодействие с самим сервисом происходит через «интерфейс» (вторая часть слова) — специальная графическая оболочка, содержащая кнопки, окна, поля для заполнения и прочих элементов.

8 пунктов, которые просто необходимы действительно хорошему интерфейсу:

* доходчивость;
* лаконичность;
* узнаваемость;
* восприимчивость;
* последовательность;
* привлекательность;
* эффективность;
* снисходительность;

## 2.2 Инструкция для пользователя

Запуск веб-приложения происходит после запуска OpenServer, выбираем пункт Консоль(Рисунок 3).

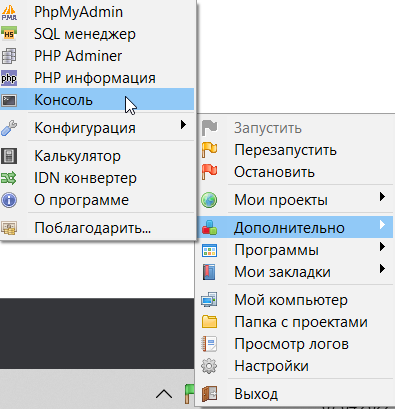


Рисунок 3 – Запуск приложения

Затем в консоль прописать команды(Рисунок 4).

cd domains

cd curstest

php artisan serve



Рисунок 4 – Выполнение команд в консоли

После запуска главное окно сайта выглядит следующим образом(Рисунок 5).

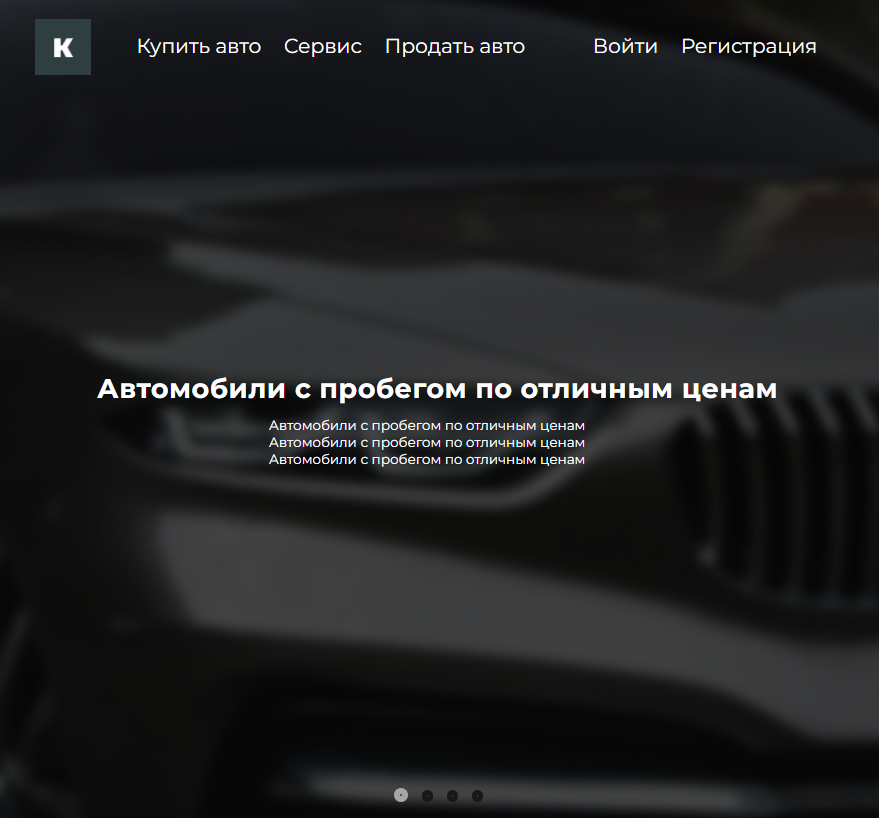


Рисунок 5 – Главное окно сайта

На главной странице есть каталог автомобилей состоящий из 3 последних добавленных автомобилей(Рисунок 6).

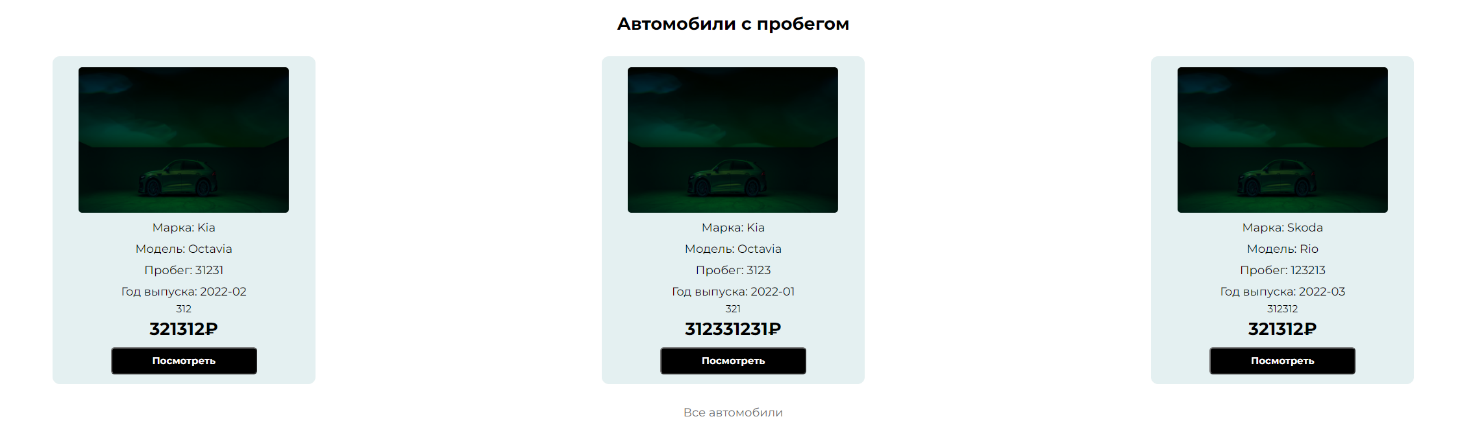


Рисунок 6 – Каталог последних добавленных автомобилей

Также на главной странице отображается форма для отправки заявки на выкуп вашего автомобиля(Рисунок 7).

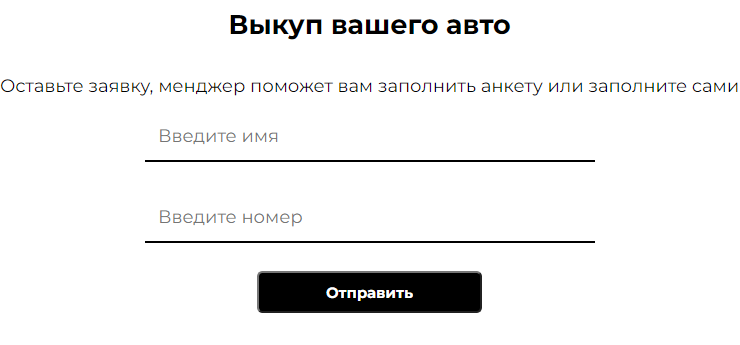


Рисунок 7 – Форма заполнения заявки на выкуп

На странице Каталога размещены все доступные автомобили(Рисунок 8).

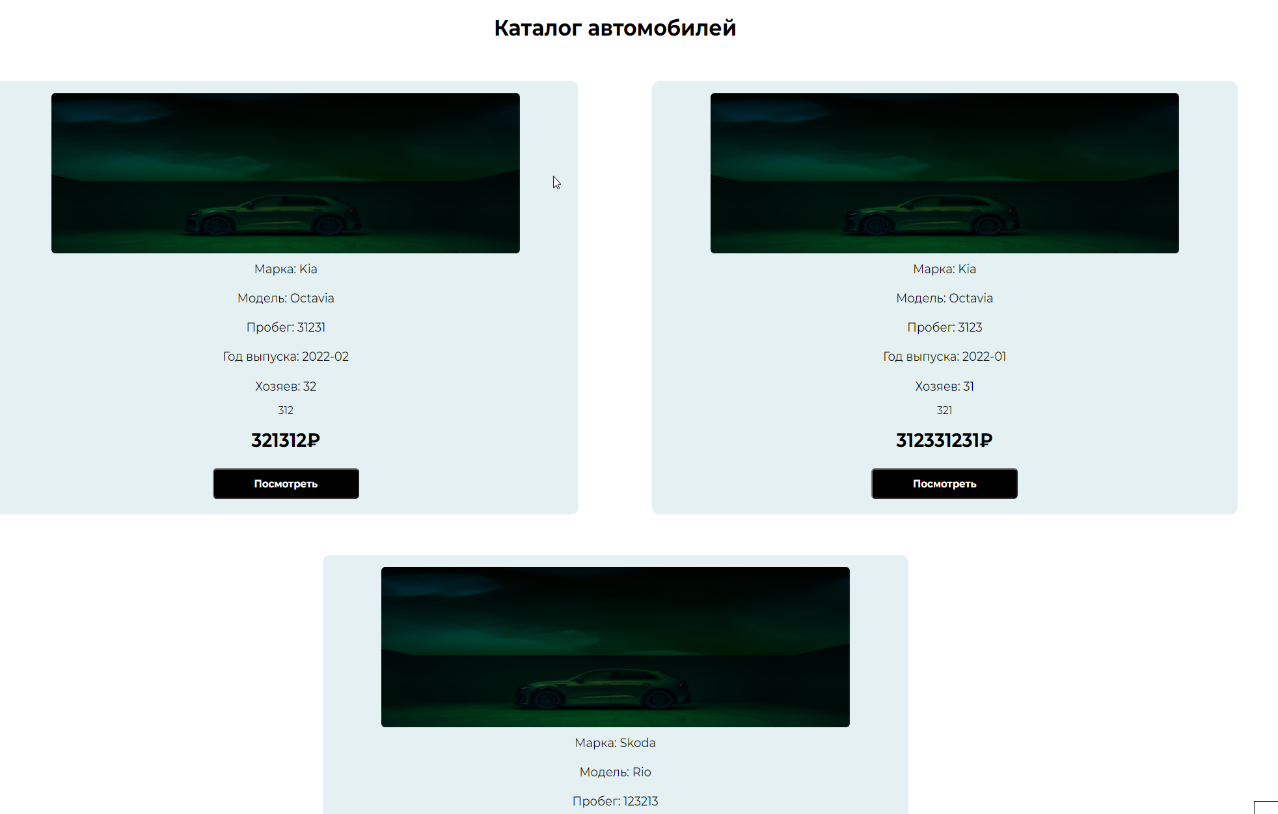


Рисунок 8 – Каталог

На странице Сервис имеется форма для отправки заявки на проведение сервисных работ(Рисунок 9).

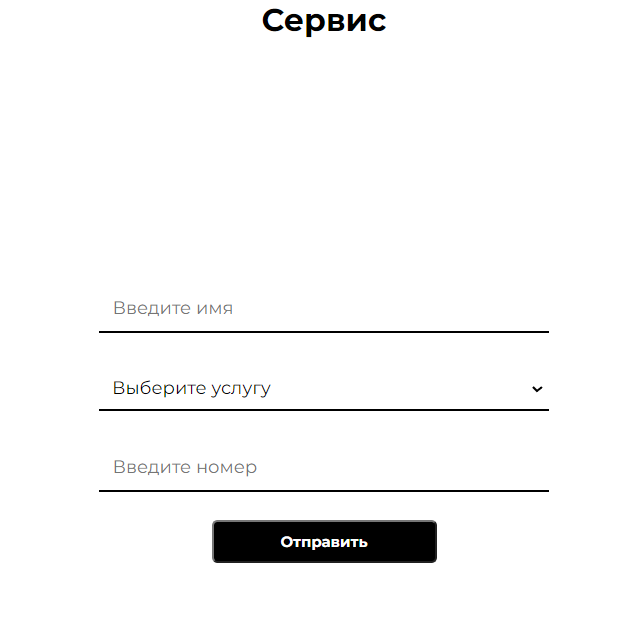


Рисунок 9 – Форма заполения заявки на сервис

Переходы в личном кабинете менеджера осуществляется через навигационное меню (Рисунок 10).

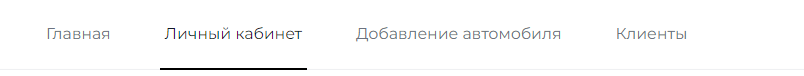


Рисунок 10 – Навигационное меню личного кабинета менеджера

Во вкладке «Добавление автомобиля» находится форма, с помощью которой можно загрузить данные в каталог(Рисунок 11).

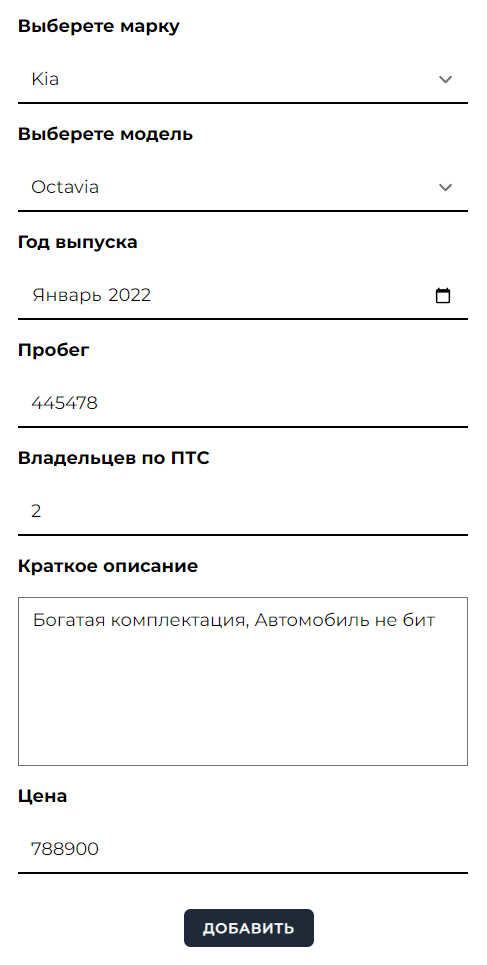


Рисунок 11 – Форма добавления автомобиля в каталог

Во вкладке «Клиенты» отображаются все клиенты оставившие заявки (Рисунок 12).

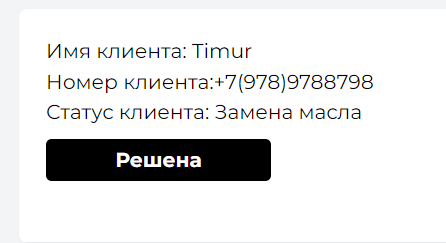


Рисунок 12 – Вывод заявки клиента

Навигационное меню в личном кабинете администратора выглядит следующим образом (Рисунок 13).

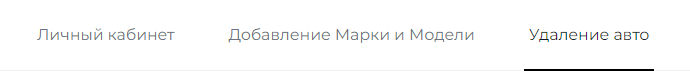


Рисунок 13 – Навигационное меню личного кабинета администратора

Во вкладке «Удаления автомобиля» можно удалить автомобиль из каталога (Рисунок 14).



Рисунок 14 – Удаления автомобиля из каталога

Во вкладке «Добавление Марки и Модели» находится форма(Рисунок 15) с её помощью можно добавить марку и модель в базу данных.

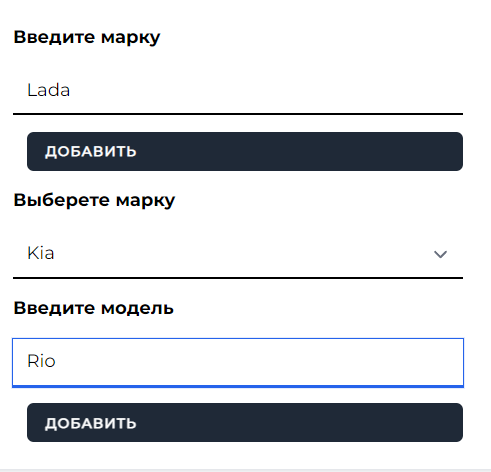


Рисунок 15 – Форма добавления марки и модели

## 2.3 Тестирование и отладка приложения

Laravel построен с учетом требований тестирования. Поддержка тестирования с помощью PHPUnit включена прямо из коробки, и файл phpunit.xml уже настроен для вашего приложения. Фреймворк также поставляется с удобными вспомогательными методами, позволяющими выразительно тестировать ваши приложения.

По умолчанию каталог tests вашего приложения содержит два каталога: Feature и Unit. Модульные (юнит) тесты – это тесты, которые фокусируются на очень небольшой изолированной части вашего кода. Фактически, большинство модульных тестов, вероятно, сосредоточены на одном методе. Тесты в каталоге «Unit» тестов не загружают ваше приложение Laravel и, следовательно, не могут получить доступ к базе данных вашего приложения или другим службам фреймворка.

Функциональные тесты могут тестировать большую часть вашего кода, включая взаимодействие нескольких объектов друг с другом, или даже целый HTTP-запрос, возвращающий JSON. Как правило, большинство ваших тестов должны быть функциональными. Эти типы тестов обеспечивают максимальную уверенность в том, что ваша система в целом работает должным образом.

Файл ExampleTest.php находится в каталогах тестов Feature и Unit. После установки нового приложения Laravel выполните команды vendor/bin/phpunit или php artisan test из командной строки для запуска ваших тестов.

В ходе создания веб-приложения были проведены тестовые запуск и исправлены все ошибки возникшие в ходе работы.

# Заключение

Информационная система - это система для поддержки принятия решений и предоставления услуг, используя компьютерную информационную технологию. Основная цель информационной системы - организация хранения поиск и передача информации.

Программное обеспечение – все или часть программ, процедур, правил и соответствующей документации системы обработки информации.

Для этого было собрана исчерпывающая информация об организации учета предоставляемых услуг салона автомобилей. После чего выявлена область для автоматизации, выбрана информация, которая должна храниться в базах данных для её лучшей обработки и анализа. После были разработаны необходимые хранилища информации, программные модули и интерфейс клиента и пользователя.

Благодаря автоматизации учета автомобилей, которые позволят оперативное сохранение данных и их реализации способствуют снижению влияния личных качеств персонала на выполнение процессов обработки документов, повышению надежности хранения, а также минимизации временных затрат на поиск и анализ необходимых изданий.

В ходе выполнения курсовой работы были закреплены теоретические знания по профессиональному модулю МДК 01.01 «Разработка программных модулей», приобретены практические навыки по веб-программированию и использования фреймворка Laravel.

Я считаю, что поставленные передо мной задачи выполнены в полном объеме.

# Список используемой литературы

1. Дакетт, Джон Основы веб-программирования с использованием HTML, XHTML и CSS / Джон Дакетт. - М.: Эксмо, 2020. - 768 c.
2. Хольцшлаг, Молли Э. Использование HTML 5 . Специальное издание (+ CD - ROM) / Хольцшлаг Молли Э.. - М.: Вильямс, 2019. - 988 c.
3. Макнейл Патрик Веб-дизайн. Книга идей веб-разработчика; Питер - Москва, 2017. - 288 c.
4. Фримен, Элизабет Изучаем HTML, XHTML и CSS / Элизабет Фримен , Эрик Фримен. - М.: Питер, 2020. - 720 c.
5. Дунаев, В. В. Базы данных. Язык SQL для студента / В.В. Дунаев. - М.: БХВ-Петербург, 2021. - 288 c.
6. Дунаев, В. Сценарии для Web-сайта. PHP и JavaScript / В. Дунаев. - М.: БХВ-Петербург, 2019. - 576 c.
7. Вотролл, Э. Изучаем веб-дизайн: учеб. пособие / Э. Вотролл, Д. Сьярто. - СПб.: Эксмо, 2018. - 496 с.
8. Гарретт, Дж. Веб-дизайн. Элементы опыта взаимодействия: учеб. пособие / Дж. Гарретт. - СПБ.: Символ-Плюс, 2020. - 192 с.
9. Глушаков, С.В. Программирование Web-страниц / С.В. Глушаков, И.А. Жакин, Т.С. Хачиров. - Харьков: «Фолио», 2017. - 390 с.
10. Глушков, С. Работа в сети Internet / С. Глушков, Д. Ломотько, А. Сурядный. - Харьков: Фолио, 2018. - 390 с.
11. Гончаров А.Ю. Web-дизайн: HTML, JavaScript и CSS. Карманный справочник / А.Ю. Гончаров. - М.: «КУДИЦ-ПРЕСС», 2017. - 320 с.
12. Документация Laravel - [Электронный ресурс]. - https://laravel.ru/
13. Современный учебник Javascript - [Электронный ресурс]. -https://learn.javascript.ru
14. Миннуллина, Р.Г. Лекционный материал по МДК 01.01 «Разработка программных модулей».

# ПРИЛОЖЕНИЯ

Листинг программы

**welcome.blade.php**

<!DOCTYPE html>

<html lang="{{ str\_replace('\_', '-', app()->getLocale()) }}">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>AS</title>

<link rel="stylesheet" href="{{ asset('css/style.css') }}">

<script src="{{ asset('js/script.js') }}" defer></script>

</head>

<body>

<div class="header slid1" id="header">

<div class="header-conteiner">

<div class="logo"><img src="img/logo.png" alt="10"></div>

<div class="menu-conteiner">

<ul class="menu">

<li class="menu-punkt"><a class="mp-main" href="{{ route('catalog') }}">Купить авто</a></li>

<li class="menu-punkt"><a class="mp-main" href="{{ route('servis') }}">Сервис</a></li>

<li class="menu-punkt"><a class="mp-main" href="#vikup">Продать авто</a></li>

</ul>

<ul class="reg">

@if (Route::has('login'))

@auth

<li class="menu-punkt"><a href="{{ url('/dashboard') }}" <a class="mp-main">Личный кабинет</a></li>

@else

<li class="menu-punkt"> <a href="{{ route('login') }}" <a class="mp-main">Войти</a></li>

@if (Route::has('register'))

<li class="menu-punkt"> <a href="{{ route('register') }}" <a class="mp-main">Регистрация</a></li>

@endif

@endauth

@endif

</ul>

</div>

</div>

<div class="header-conteiner-tittle">

<div class="header-tittle">

Автомобили с пробегом по отличным ценам

</div>

<div class="header-text">

Автомобили с пробегом по отличным ценам

</div>

</div>

<div class="slider-conteiner">

<div class="switch">

<input type="radio" name="elem" class="radiobtn" id="rad1" value="1" checked>

<input type="radio" name="elem" class="radiobtn" id="rad2" value="2">

</div>

</div>

</div>

</div>

</div>

<div class="main">

<div class="main-conteiner">

<div class="uslg">

<div class="uslg-conteiner">

<div class="uslg-title">Наш салон автомобилей предлагает следующие услуги</div>

<div class="uslg-box">

<div class="uslg-boxs">Купить автомобиль с пробегом</div>

<div class="uslg-boxs">Продать ваш автомобиль</div>

<!-- <div class="uslg-boxs">Оформить полис ОСАГО/КАСКО</div> -->

<div class="uslg-boxs">Провести ТО</div>

</div>

</div>

</div>

<div class="catalog\_avto">

<div class="catalog\_avto-conteiner">

<div class="uslg-title">

Автомобили с пробегом

</div>

<div class="catalog-box">

@foreach($car as $el)

<div class="catalog-item">

<div class="item-image"><img src="img/bg-header.png" alt="1"></div>

<div class="item-t">Марка: {{$el->cars->addbrand}} </div>

<div class="item-t">Модель: {{$el->model}}</div>

<div class="item-t">Пробег: {{$el->probeg}}</div>

<div class="item-t">Год выпуска: {{$el->year}}</div>

<div class="item-descript">{{$el->opis}}</div>

<div class="item-price">{{$el->price}}₽</div>

<div class="btn-box"><button class="item-btn">Посмотреть</button></div>

</div>

@endforeach

</div>

<div class="catalog-link">

<a href="{{ route('catalog') }}">Все автомобили</a>

</div>

</div>

</div>

<div class="catalog\_servis">

<div class="catalog\_servis-conteiner">

<div class="uslg-title">Сервис</div>

<div class="uslg-box">

<div class="uslg-boxs">Купить автомобиль с пробегом</div>

<div class="uslg-boxs">Продать ваш автомобиль</div>

<div class="uslg-boxs">Оформить полис ОСАГО/КАСКО</div>

<div class="uslg-boxs">Провести ТО</div>

</div>

<div class="catalog-link">

<a href="{{ route('servis') }}">Все услуги</a>

</div>

</div>

</div>

<div class="form\_manager">

<div class="form\_manager-conteiner">

<div class="uslg-title">Выкуп вашего авто</div>

<div class="catalog-link"> Оставьте заявку, менджер поможет вам заполнить анкету или заполните сами</div>

<div class="uslg-box">

<div class="form-cont" id="vikup">

<form action="{{ route('vikup-submit') }}" method="post" class="form-post">

@csrf

<input type="text" name="fio\_client" id="fio\_client" class="inp-add inps" placeholder="Введите имя" required>

<input type="text" name="number\_client" id="number\_client" class="inp-add inps" placeholder="Введите номер" required>

<input type="hidden" name="status\_client" id="status\_client" value="Выкуп авто">

<div class="btn-box"><button class="item-btn">Отправить</button></div>

</form>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<script src="{{ asset('js/jq.js') }}" defer></script>

<script src="{{ asset('js/jquery.maskedinput.js') }}" defer></script>

<script src="{{ asset('js/mask.js') }}" defer></script>

<script src="{{ asset('js/testajax.js') }}" defer></script>

</body>

</html>

**testajax.js**

$(document).ready(() => {

$("#model").hide();

$("#model-label").hide();

$("#brand").change(function() {

if($('#brand').val() == 0){

$("#model").hide("slow" );

$("#model-label").hide("slow" );}

else{

$("#model-label").show("slow" );

$("#model").show("slow" );

}

var brand\_id = $(this).val();

$.ajax({

url: "/addCar",

type: "GET",

data: {

brand\_id: brand\_id

},

headers: {

'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')

},

success: (data) => {

$('#model').html(data);

}

});

})

});

**create\_cars\_table.php**

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateCarsTable extends Migration

{

public function up()

{

Schema::create('cars', function (Blueprint $table) {

$table->id();

$table->foreignId('brand\_id')->constrained();

$table->string('model');

$table->string('year');

$table->string('probeg');

$table->string('xoz');

$table->string('opis');

$table->string('price');

$table->timestamps();

});

}

public function down()

{

Schema::dropIfExists('cars');

}

}

**web.php**

<?php

use Illuminate\Support\Facades\Route;

Route::get('/', 'carController@welcomePaginate')->name('/');

Route::post('/submit', 'vikupController@vikupSubmit')->name('vikup-submit');

Route::post('/uslg-submit', 'vikupController@uslgSubmit')->name('uslg-submit');

Route::post('/catalog-submit', 'vikupController@zayvkaSubmit')->name('zayvka-submit');

Route::get('/dashboard', function () {return view('dashboard');})->middleware(['auth'])->name('dashboard');

Route::get('/addCar', 'brandController@dateBrand')->middleware(['auth'])->name('addCar');

Route::post('/addCar/submit', 'carController@carSubmit')->middleware(['auth'])->name('car-submit');

Route::get('/addBrand', 'brandController@dbBrand')->middleware(['auth', 'access:admin'])->name('addBrand');

Route::post('/addBrand/submit', 'brandController@addCar')->middleware(['auth'])->name('brand-submit');

Route::post('/addModel/submit', 'brandController@addModel')->middleware(['auth'])->name('model-submit');

Route::get('/deleteCar', 'carController@delcatalogDate')->middleware(['auth'])->name('deleteCar');

Route::get('/deleteCar/{id}/delete', 'carController@delcatalogDelete')->middleware(['auth'])->name('car-delete');

Route::get('/clients', 'vikupController@dateClient')->middleware(['auth'])->name('clients');

Route::get('/clients/{id}/delete', 'vikupController@delClient')->middleware(['auth'])->name('deleteClient');

Route::get('/catalog', 'carController@catalogDate')->name('catalog');

Route::get('/servis', function () {return view('servis');})->name('servis');

require \_\_DIR\_\_ . '/auth.php';

**Brand.php**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Brand extends Model

{

public function avtoMs()

{

return $this->hasMany(avtoM::class);

}

}

**brandController.php**

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\Brand;

use App\Models\avtoM;

class brandController extends Controller

{

public function addCar(Request $req)

{

$zayvka = new Brand;

$zayvka->addbrand = $req->input('addbrand');

$zayvka->save();

return redirect()->route('addBrand');

}

public function addModel(Request $req)

{

$zayvka = new avtoM;

$zayvka->models = $req->input('addmodel');

$zayvka->brand\_id = $req->input('brands');

$zayvka->save();

return redirect()->route('addBrand');

}

public function dateBrand(Request $req)

{

$brand = Brand::all();

$model = 0;

if (isset($req->brand\_id)) {

$check = $req->brand\_id;

$model = Brand::find($check)->avtoMs;

}

if($req -> ajax()){

return view('ajax.models',

['model' => $model])->render();

}

return view('addCar', ['brand' => $brand,'model' => $model]);

}

public function dbBrand()

{

return view('addBrand', ['brand' => Brand::all()]);

}

}