

Министерство образования Иркутской области

Государственное бюджетное профессиональное образовательное учреждение

Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ДП. 09.02.07-3.25.211.12. ПЗ

УТВЕРЖДАЮ

Зам. директора по УР, к.т.н.

_____ Е.А. Коробкова

ВЕБ-ПРИЛОЖЕНИЕ «АВТОСАЛОН»

Нормоконтролер:

(подпись, дата)

(Е.С. Кудрявцева)

Консультат по

(подпись, дата)

(А.П. Юргина)

экономической части:

Руководитель:

(подпись, дата)

(А. П. Стош)

Студент:

(подпись, дата)

(К. С. Мясников)

Иркутск 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 Предпроектное исследование	5
1.1 Описание предметной области.....	5
2 Техническое задание.....	9
3 Проектирование программного продукта	10
3.1 Архитектура программного продукта	10
3.2 Функциональное и структурное проектирование	11
3.3 Проектирование базы данных	13
3.4 Проектирование пользовательского интерфейса	23
3.4.1 Разработка прототипов интерфейса.....	23
3.4.2 Выбор цветовой гаммы и шрифтов	28
3.4.3 Разработка элементов интерфейса.....	28
3.4.4 Разработка дизайн макетов.....	30
4 Реализация программного продукта	33
4.1 Разработка клиентской части.....	33
4.2 Разработка базы данных	35
4.3 Разработка серверной части	37
5 Отладка и тестирование программного продукта	39
6 Документация программного продукта	43
6.1 Руководство пользователя.....	43
6.2 Руководство администратора.....	49
6.3 Руководство менеджера	54
7 Стоимость разработки.....	56
7.1 Расчет трудоемкости разработки программного обеспечения.....	56
7.2 Затраты на оплату.....	57
7.3 Затраты на амортизацию оборудования	58
7.4 Расчет затрат на электроэнергию	59
7.5 Затраты на материалы, израсходованные при проведении разработки	60
7.6 Расчет затрат на разработку программного обеспечения.....	60
7.7 Расчет цены.....	61
ЗАКЛЮЧЕНИЕ	62
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	63
Приложение А Техническое задание	65
Приложение Б код клиентской части.....	72
Приложение В код серверной части.....	75

					ДП.09.02.07-3.25.211.12. ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.		Мясников К.С..			ВЕБ-ПРИЛОЖЕНИЕ «АВТОСАЛОН» пояснительная записка	Лит.	Лист	Листов
Провер.		Стош П.А.					2	80
Реценз.						ГБПОУИО «ИАТ» ВЕБ-21-1		
Н.контр.		Кудрявцева Е.С.						
Утверд.		Коробкова Е.А.						

ВВЕДЕНИЕ

Автосалон – это коммерческая организация, занимающаяся продажей новых и поддержанных автомобилей. В современном мире, особенно в условиях роста городской инфраструктуры и увеличения числа владельцев транспортных средств, спрос на услуги автосалонов продолжает расти. Это связано с изменением потребительских привычек, развитием цифровых технологий и стремлением к удобству при совершении крупных покупок. Многие потенциальные покупатели всё чаще предпочитают заранее ознакомиться с ассортиментом через интернет, чтобы сэкономить время и сделать более осознанный выбор.

Для бизнеса продажа автомобилей — это не только вопрос получения прибыли, но и важный элемент в формировании имиджа компании, уровня доверия клиентов и конкурентоспособности на рынке. Удобство выбора авто, наличие подробной информации и возможность оформления заявки онлайн положительно влияют на лояльность покупателей и повышают эффективность работы автосалона.

Спектр возможностей, предоставляемых автосалонами, постоянно расширяется и адаптируется под запросы клиентов. Среди наиболее востребованных функций можно выделить: просмотр каталога автомобилей с детальными характеристиками, фильтрация по параметрам (цена, марка, модель, пробег), возможность оставить заявку на интересующую машину. Такие решения делают процесс покупки автомобиля максимально простым и доступным как для частных лиц, так и для организаций.

В условиях высокой конкуренции среди автосалонов автоматизация процессов становится важным инструментом для повышения эффективности работы и улучшения клиентского сервиса. Использование веб-приложений позволяет оперативно обновлять информацию о наличии автомобилей, упрощает взаимодействие с клиентами, сокращает время на обработку заявок и повышает уровень удовлетворённости пользователей. Это способствует не только

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						3
Изм.	Лист	№ докум.	Подпись	Дата		

привлечению новых клиентов, но и укреплению доверия со стороны постоянных покупателей, что является залогом успешного развития бизнеса.

Таким образом, автосалоны играют важную роль в автомобильной индустрии, предлагая комплексное решение для подбора и покупки автомобиля. Они экономят время клиентов, предоставляют полную информацию о транспортных средствах и обеспечивают высокое качество обслуживания, что делает их незаменимыми партнерами при приобретении личного авто.

Целью дипломного проектирования является разработка веб-приложения автосалона, которое позволит автоматизировать процессы отображения и выбора автомобилей, а также приема заявок от клиентов.

Основными задачами являются:

- произвести исследование деятельности автосалона;
- определить инструментальные средства проектирования и реализации;
- разработать техническое задание для веб-приложения;
- спроектировать базу данных;
- спроектировать интерфейс пользователя веб-приложения;
- реализовать веб-приложение;
- провести тестирование веб-приложения;
- составить программную документацию.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

1 Предпроектное исследование

1.1 Описание предметной области

В качестве предметной области была выбрана автоматизация процессов в автосалоне, специализирующемся на продаже новых и поддержанных автомобилей. Основной задачей разработки данного веб-приложения является оптимизация процессов взаимодействия между клиентами, менеджерами и администратором, упрощение процедуры выбора и бронирования автомобиля, а также обеспечение удобного и понятного интерфейса для всех участников процесса.

Веб-приложение должно автоматизировать ключевые этапы работы автосалона: от просмотра каталога автомобилей до оформления заявки на покупку. Это позволит сократить время на обработку запросов, повысить прозрачность информации и улучшить качество обслуживания клиентов.

Работа администратора заключается в управлении данными о автомобилях (добавление, редактирование, удаление), а также в управлении учетными записями менеджеров.

Менеджер может просматривать список автомобилей, информацию по каждому из них, а также список оформленных клиентами заявок и бронирований. Он отвечает за редактирование статуса авто и статуса заявки клиента.

Клиент имеет возможность зарегистрироваться, просмотреть каталог автомобилей, выбрать интересующий вариант, оставить заявку на покупку автомобиля, а также просмотреть историю своих заявок и бронирований.

Для эффективной работы веб-приложения будет разработан интуитивно понятный интерфейс как для клиентов, так и для сотрудников автосалона.

В соответствии с предметной областью можно выделить следующие объекты:

– Автомобиль. Атрибуты: код автомобиля, vin номер, марка, модель, поколение, комплектация, год выпуска, пробег, стоимость, статус (в

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

наличии/продан), описание, фото, цвет, тип кузова, тип двигателя, мощность, тип трансмиссии, привод, страна производства, запас хода, трехмерная модель, автосалон.

– Клиент. Атрибуты: код клиента, имя, контактные данные, история бронирований.

– Менеджер. Атрибуты: код менеджера, имя, контактные данные, список бронирований клиентов.

– Бронирование. Атрибуты: код бронирования, код клиента, код автомобиля, дата визита в автосалон, дата бронирования, комментарий менеджера, статус (ожидание/подтвержденное/отменено/выполненное).

Веб-приложение создается для обслуживания следующих категорий пользователей:

– Администратор - управляет автомобилями, менеджерами и филиалами.

– Менеджер - работает с информацией о бронированиях автомобилей, обрабатывает бронирования.

– Клиент - регистрируется, просматривает каталог автомобилей, оставляет заявки, отслеживает статусы.

Таким образом, веб-приложение для автосалона должно стать надежным инструментом для повышения эффективности работы, улучшения качества обслуживания клиентов и увеличения скорости продаж автомобилей.

1.2 Обзор инструментов, используемых в проектировании и разработке программного обеспечения

Для проектирования и разработки веб-приложения были выбраны современные инструменты и технологии, которые обеспечивают эффективную реализацию всех этапов проекта.

Draw.io - это удобный онлайн-инструмент для построения диаграмм. Он предоставляет широкий набор возможностей для визуального представления

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

различных процессов и структур, что особенно важно на этапе проектирования архитектуры приложения.

Figma - бесплатная платформа для создания дизайна и прототипирования пользовательских интерфейсов. С её помощью можно разработать наглядный и функциональный макет веб-приложения ещё до начала программирования.

MySQL Workbench - мощный инструмент для работы с базами данных. Он позволяет не только управлять данными, но и создавать ER-диаграммы, отражающие структуру таблиц и связи между ними. Этот инструмент будет использован в рамках дипломного проекта для проектирования базы данных автосалона.

Для реализации клиентской части веб-приложения выбраны следующие технологии:

HTML - язык разметки, используемый для построения структуры веб-страниц. Он определяет расположение элементов на странице, но не влияет на их внешний вид.

Tailwind CSS - популярный фреймворк для стилизации веб-страниц. Он позволяет быстро и гибко создавать современный дизайн, применяя классы непосредственно в HTML.

JavaScript - язык программирования, применяемый для добавления интерактивности к веб-страницам. Он обеспечивает динамическое обновление контента, обработку событий и взаимодействие с сервером.

Создание надёжной серверной части является важной частью разработки веб-приложения, поскольку она отвечает за обработку запросов, работу с базой данных и безопасность системы.

PHP - один из самых популярных языков программирования для создания серверной логики. Он хорошо подходит для работы с базами данных и позволяет строить масштабируемые веб-приложения.

Для ускорения и упрощения разработки используется Laravel - современный PHP-фреймворк, предоставляющий готовые решения для маршрутизации,

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

аутентификации, управления сессиями и других задач. Использование Laravel повышает качество кода и снижает риск возникновения ошибок.

Для хранения и управления данными выбрана система MySQL - реляционная СУБД, в которой данные организованы в виде таблиц. Она отличается высокой производительностью, хорошей документацией и активным сообществом разработчиков.

В качестве среды разработки используется Visual Studio Code (VS Code) - мощный и гибкий редактор кода, поддерживающий множество расширений. Он позволяет удобно работать с файлами, настраивать автоматизацию и писать чистый, структурированный код.

Для контроля версий и совместной работы над проектом используются:

Git - распределённая система контроля версий, которая помогает отслеживать изменения в коде, создавать отдельные ветки для разных задач и восстанавливать предыдущие версии проекта при необходимости. Git обеспечивает стабильное управление кодовой базой при командной разработке.

GitLab - веб-платформа, позволяющая размещать и управлять репозиториями на основе Git. Платформа предоставляет дополнительные возможности для автоматизации тестирования и развертывания приложений, а также систему управления задачами. Это делает рабочий процесс более прозрачным и организованным.

Выбранные инструменты и технологии обеспечивают надежную реализацию, простоту сопровождения и защиту кодовой базы веб-приложения на всех этапах его жизненного цикла.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

2 Техническое задание

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 34.602-2020.

Согласно ГОСТ 34.602-2020 техническое задание должно включать следующие разделы:

- 1) введение;
- 2) основания для разработки;
- 3) назначение системы;
- 4) требования к системе;
- 5) требования к техническому обеспечению;
- 6) требования к программному обеспечению;
- 7) организационно-технические требования.

Техническое задание на разработку веб-приложения «Автосалон» представлено в приложении А.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

3 Проектирование программного продукта

3.1 Архитектура программного продукта

В веб-приложении используется клиент-серверная архитектура. Данная архитектура делится на две основные части: клиентскую и серверную.

Клиентская часть представляет собой интерфейс, с помощью которого пользователь взаимодействует с системой, пользователь отправляет HTTP-запросы к серверу и получает ответ.

Серверная часть обрабатывает запросы, взаимодействует с базой данных и возвращает ответ.

Основные принципы:

- Разделение обязанностей: клиент отвечает за интерфейс и отправку запросов, сервер – за обработку и хранение данных.
- Сетевое взаимодействие: обмен данными происходит по стандартным протоколам (HTTP, TCP/IP, WebSocket).
- Централизованное управление: сервер часто выступает единой точкой контроля данных и бизнес-логики.

Так же используется архитектурный паттерн MVC (Model-View-Controller), который разделяет веб-приложение на три основных компонента:

- Model (Модель) отвечает за работу с базой данных.
- View (Представление) отображает данные пользователю.
- Controller (Контроллер) является посредником между моделью и представлением. Обрабатывает входящие запросы, взаимодействует с моделью для получения данных и передает их в представление.

Особенности клиент-серверной архитектуры:

- Сервер может быть легко масштабирован для обработки большого количества запросов.
- Клиентская часть может быть оптимизирована для различных устройств.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						10
Изм.	Лист	№ докум.	Подпись	Дата		

3.2 Функциональное и структурное проектирование

Диаграмма прецедентов — это один из типов диаграмм в языке UML (Unified Modeling Language), используемый для моделирования функциональных требований системы. Она помогает наглядно показать, как различные актеры взаимодействуют с системой, какие функции они могут использовать и при каких условиях. Такая диаграмма широко применяется на этапе анализа и проектирования программного обеспечения для уточнения и документирования поведенческих аспектов системы.

В веб-приложении выделяются три ролей: администратор, пользователь, менеджер.

На основе этих ролей была разработана диаграмма прецедентов, которая включает все ключевые функции системы. Она наглядно демонстрирует: какие действия доступны каждой роли, как взаимодействуют между собой разные участники системы, какие сценарии использования реализованы в веб-приложении

1) Пользователь: регистрация и авторизация в системе, возможность осуществлять поиск, оформление бронирования на покупку.

2) Администратор: добавление новых менеджеров, управление автомобилями.

3) Менеджер: рассмотрение и редактирование бронирования клиента.

Итогом является то, что диаграмма прецедентов (рисунок 1) должна отражать ключевые действия, которые пользователи выполняют в рамках информационной системы, а также демонстрировать способы их взаимодействия как с системой, так и между собой. Такая диаграмма позволяет наглядно представить функциональные возможности системы с точки зрения её конечных пользователей, выделить роли, участвующие в процессах, и показать, как различные участники взаимодействуют друг с другом через выполняемые действия.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

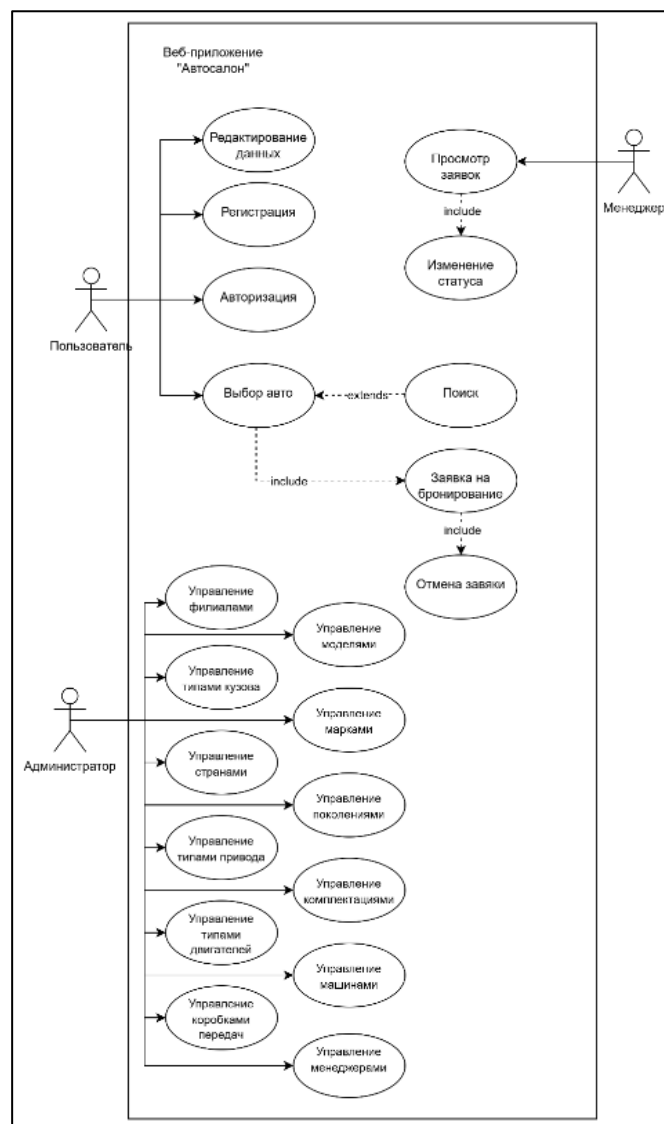


Рисунок 1 – Диаграмма прецедентов

Диаграмма деятельности представляет собой визуальную модель в виде блок-схемы, которая демонстрирует последовательность действий и показывает, как поток управления перемещается от одной операции к другой внутри системы. На рисунке 2 с помощью диаграммы деятельности изображен процесс, который проходит пользователь при оформлении заявки на бронирование. Такая диаграмма служит важным инструментом для анализа и оптимизации бизнес-процессов, а также помогает в понимании поведения пользователя в рамках веб-приложения.

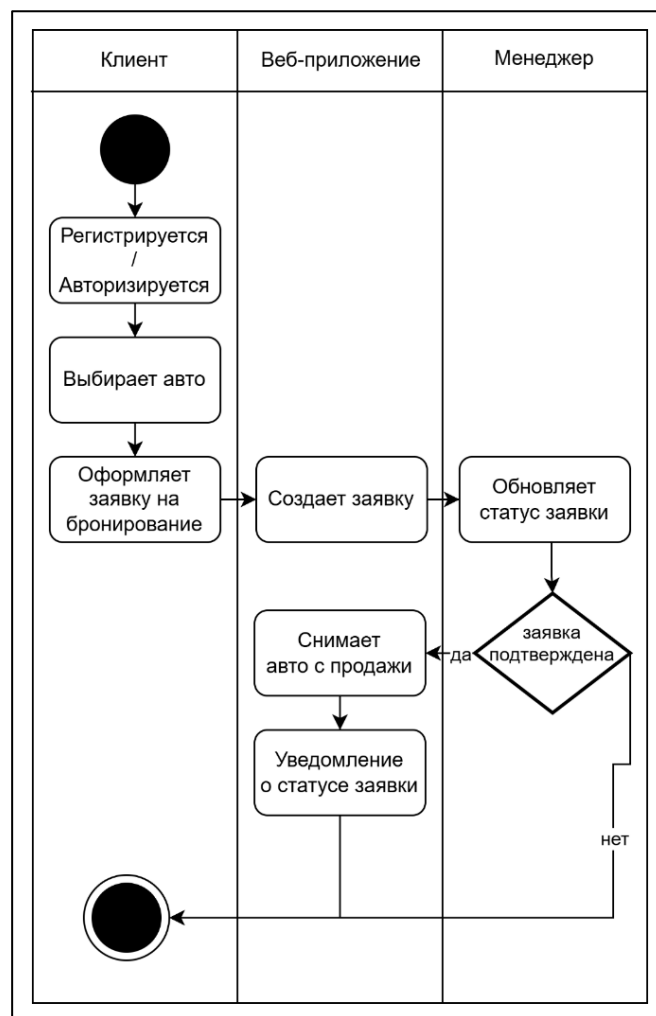


Рисунок 2 – Диаграмма деятельности

Диаграмма представляет блок-схему деятельности, демонстрируя последовательность действий от регистрации клиента до оформления бронирования.

3.3 Проектирование базы данных

База данных – это структурированный набор упорядоченной информации, которая храниться в связанных электронных таблицах.

ER-модель – это концептуальная модель взаимосвязей объектов с набором атрибутов, позволяющая визуализировать то, как объекты связаны. ER-модель веб-приложения представлена на рисунке 3.

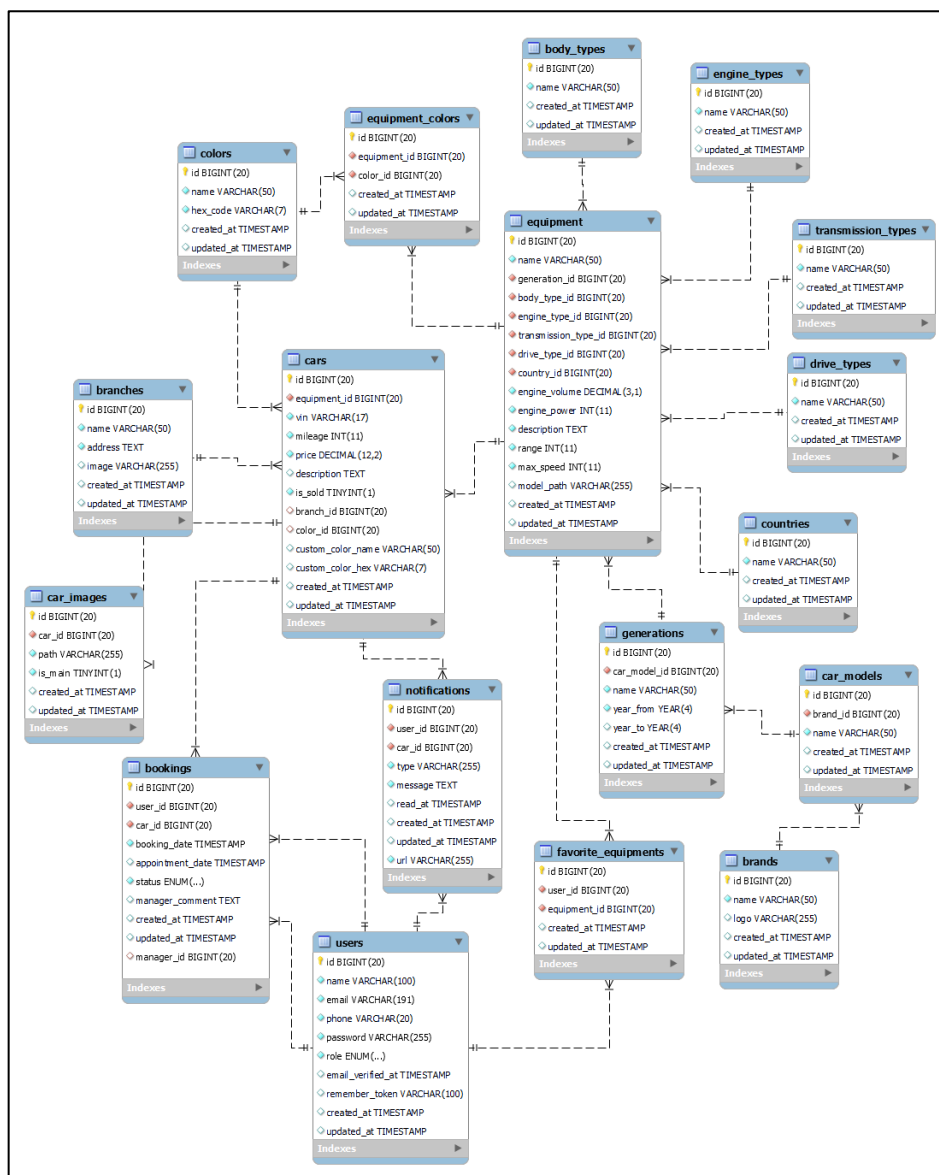


Рисунок 3 – ER-модель веб-приложения

Нормальные формы – это набор правил, которые помогают улучшить структуру базы данных

Все таблицы представлены в третьей нормальной форме (3НФ), что означает, что каждая таблица удовлетворяет условиям 1НФ и 2НФ, а также не имеет транзитивных зависимостей. Каждый столбец содержит атомарное значение, не ключевые атрибуты зависят от каждого потенциального ключа, и нет транзитивных зависимостей от первичного ключа.

Далее в таблицах 1-18 представлены таблицы ER модели, их поля, типы данных и описание.

Таблица 1 – Users (пользователи)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор пользователя
name	VARCHAR(100)	Имя пользователя
email	VARCHAR(191) UNIQUE	Адрес электронной почты
phone	VARCHAR(20) UNIQUE	Номер телефона
password	VARCHAR(255)	Хэшированный пароль
role	ENUM('admin', 'manager', 'user') DEFAULT 'user'	Роль пользователя
email_verified_at	TIMESTAMP NULLABLE	Дата и время подтверждения email
remember_token	VARCHAR(100) NULLABLE	Токен для "Remember me"
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 2 – Brands (бренды)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название бренда
logo	VARCHAR(255) NULLABLE	Путь к изображению логотипа
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 3 – Countries (страны)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название страны

Продолжение таблицы 3

Поле	Тип данных	Описание
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 4 – Body_types (типы кузова)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название типа кузова
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 5 – Car_models (модели авто)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
brand_id (FK)	BIGINT FOREIGN KEY → brands.id	Внешний ключ бренда
name	VARCHAR(50)	Название модели автомобиля
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 6 – Generations (поколения)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
car_model_id (FK)	BIGINT FOREIGN KEY → car_models.id	Внешний ключ модели автомобиля
name	VARCHAR(50)	Название поколения
year_from	YEAR	Год начала выпуска
year_to	YEAR NULLABLE	Год окончания выпуска
year_from	YEAR	Год начала выпуска

Продолжение таблицы 6

Поле	Тип данных	Описание
year_from	YEAR	Год начала выпуска
year_to	YEAR NULLABLE	Год окончания выпуска
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 7 – Drive_types (типы привода)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название привода
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 8 – Engine_types (типы двигателей)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название типа двигателя
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 9 – Transmission_types (типы трансмиссий)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название трансмиссии
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 10 – Colors (цвета)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название цвета
hex_code	VARCHAR(7)	HEX-код цвета
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 11 – Equipment (комплектации)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
name	VARCHAR(50)	Название комплектации
generation_id (FK)	BIGINT FOREIGN KEY → generations.id	Внешний ключ поколения
body_type_id (FK)	BIGINT FOREIGN KEY → body_types.id	Внешний ключ типа кузова
engine_type_id (FK)	BIGINT FOREIGN KEY → engine_types.id	Внешний ключ типа двигателя
engine_volume	DECIMAL(3,1)	Объем двигателя (литры)
engine_power	INTEGER	Мощность двигателя (л.с.)
transmission_type_id (FK)	BIGINT FOREIGN KEY → transmission_types.id	Внешний ключ типа трансмиссии
drive_type_id (FK)	BIGINT FOREIGN KEY → drive_types.id	Внешний ключ типа привода
country_id (FK)	BIGINT FOREIGN KEY → countries.id	Внешний ключ страны производства
description	TEXT NULLABLE	Описание комплектации
range	INTEGER	Запас хода (км)
max_speed	INTEGER	Максимальная скорость (км/ч)
range	INTEGER	Запас хода (км)

Продолжение таблицы 11

Поле	Тип данных	Описание
max_speed	INTEGER	Максимальная скорость (км/ч)
model_path	VARCHAR(255) NULLABLE	Путь для 3d модели
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 12 – Equipments_colors (цвета комплектаций)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
equipment_id (FK)	BIGINT FOREIGN KEY → equipment.id	Внешний ключ комплектации
color_id (FK)	BIGINT FOREIGN KEY → colors.id	Внешний ключ цвета
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 13 – Branches (Филиалы)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор филиала
name	VARCHAR(50)	Название филиала
address	TEXT	Адрес филиала
image	VARCHAR(255) NULLABLE	Путь к изображению филиала
created_at	TIMESTAMP	Дата создания записи
name	VARCHAR(50)	Название филиала
address	TEXT	Адрес филиала
image	VARCHAR(255) NULLABLE	Путь к изображению филиала
name	VARCHAR(50)	Название филиала
address	TEXT	Адрес филиала

Продолжение таблицы 13

Поле	Тип данных	Описание
image	VARCHAR(255) NULLABLE	Путь к изображению филиала
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 14 – Cars (Авто)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор автомобиля
equipment_id (FK)	BIGINT FOREIGN KEY → equipment.id	Внешний ключ комплектации
vin	VARCHAR(17) UNIQUE	VIN-номер автомобиля
mileage	INTEGER	Пробег (в км)
price	DECIMAL(12, 2)	Цена автомобиля
description	TEXT NULLABLE	Описание автомобиля
is_sold	BOOLEAN DEFAULT false	Статус продажи авто
branch_id (FK)	BIGINT FOREIGN KEY → branches.id NULLABLE	Внешний ключ филиала (может быть NULL)
color_id (FK)	BIGINT FOREIGN KEY → colors.id NULLABLE	Внешний ключ цвета автомобиля (может быть NULL)
custom_color_name	VARCHAR(50) NULLABLE	Название пользовательского цвета
custom_color_hex	VARCHAR(7) NULLABLE	HEX-код пользовательского цвета
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 15 – Car_images (изображения авто)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор изображения
car_id (FK)	BIGINT FOREIGN KEY → cars.id	Внешний ключ автомобиля
path	VARCHAR(255)	Путь к файлу изображения
is_main	BOOLEAN DEFAULT false	Статус основного изображения
created_at	TIMESTAMP	Дата добавления изображения
updated_at	TIMESTAMP	Дата последнего обновления

Таблица 16 – Bookings (бронирования)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор брони
user_id (FK)	BIGINT FOREIGN KEY → users.id	Внешний ключ пользователя
car_id (FK)	BIGINT FOREIGN KEY → cars.id	Внешний ключ авто
booking_date	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Дата бронирования
appointment_date	TIMESTAMP	Дата визита в автосалон
status	ENUM('pending', 'confirmed', 'rejected', 'completed')	Статус бронирования
manager_comment	TEXT NULLABLE	Комментарий менеджера
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления
INDEX	INDEX(car_id, status)	Для оптимизации поиска по статусам и авто

Таблица 17 – Favorite equipments (избранные комплектации)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор
user_id (FK)	BIGINT FOREIGN KEY → users.id	Внешний ключ пользователя
equipment_id (FK)	BIGINT FOREIGN KEY → equipment.id	Внешний ключ комплектации
created_at	TIMESTAMP	Дата добавления
updated_at	TIMESTAMP	Дата последнего обновления
UNIQUE	UNIQUE(user_id, equipment_id)	Одна пара пользователь/комплектация может быть только одна

Таблица 18 – Notifications (уведомления)

Поле	Тип данных	Описание
id (PK)	BIGINT UNSIGNED AUTO_INCREMENT	Уникальный идентификатор уведомления
user_id (FK)	BIGINT FOREIGN KEY → users.id	Кому отправлено уведомление
car_id (FK)	BIGINT FOREIGN KEY → cars.id	С каким автомобилем связано уведомление
type	VARCHAR	Тип уведомления
message	TEXT	Текст уведомления
read_at	TIMESTAMP NULLABLE	Время прочтения уведомления
created_at	TIMESTAMP	Дата создания записи
updated_at	TIMESTAMP	Дата последнего обновления
url	VARCHAR	Ссылка для перехода

3.4 Проектирование пользовательского интерфейса

3.4.1 Разработка прототипов интерфейса

Для того что бы наглядно показать, как страницы связаны между собой на рисунке 4 представлена схема навигации.

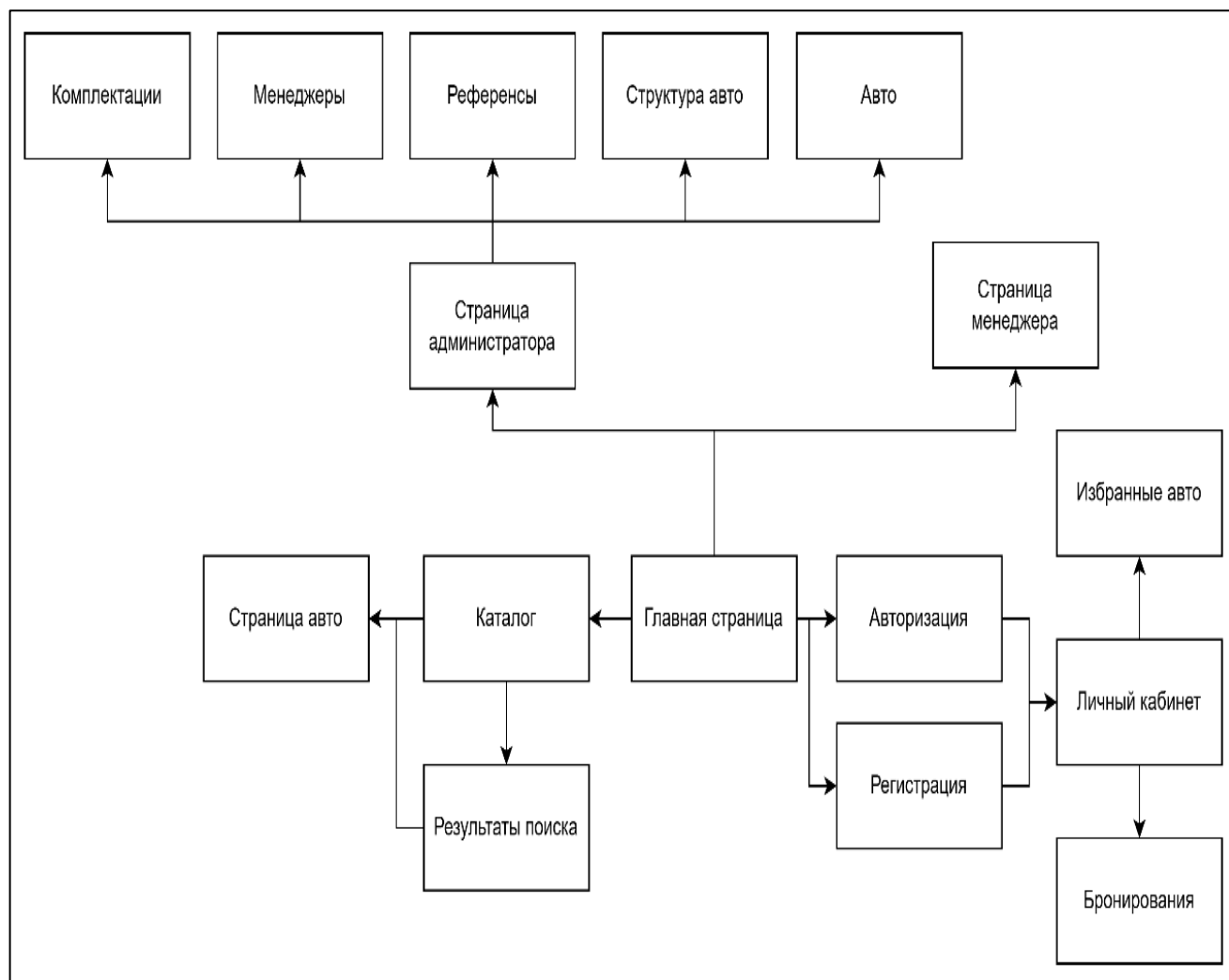


Рисунок 4 – Схема навигации

Для разработки пользовательского интерфейса используется онлайн-сервис «Figma». На рисунках 5 – 9 представлены прототипы страниц веб-приложения автосалона.

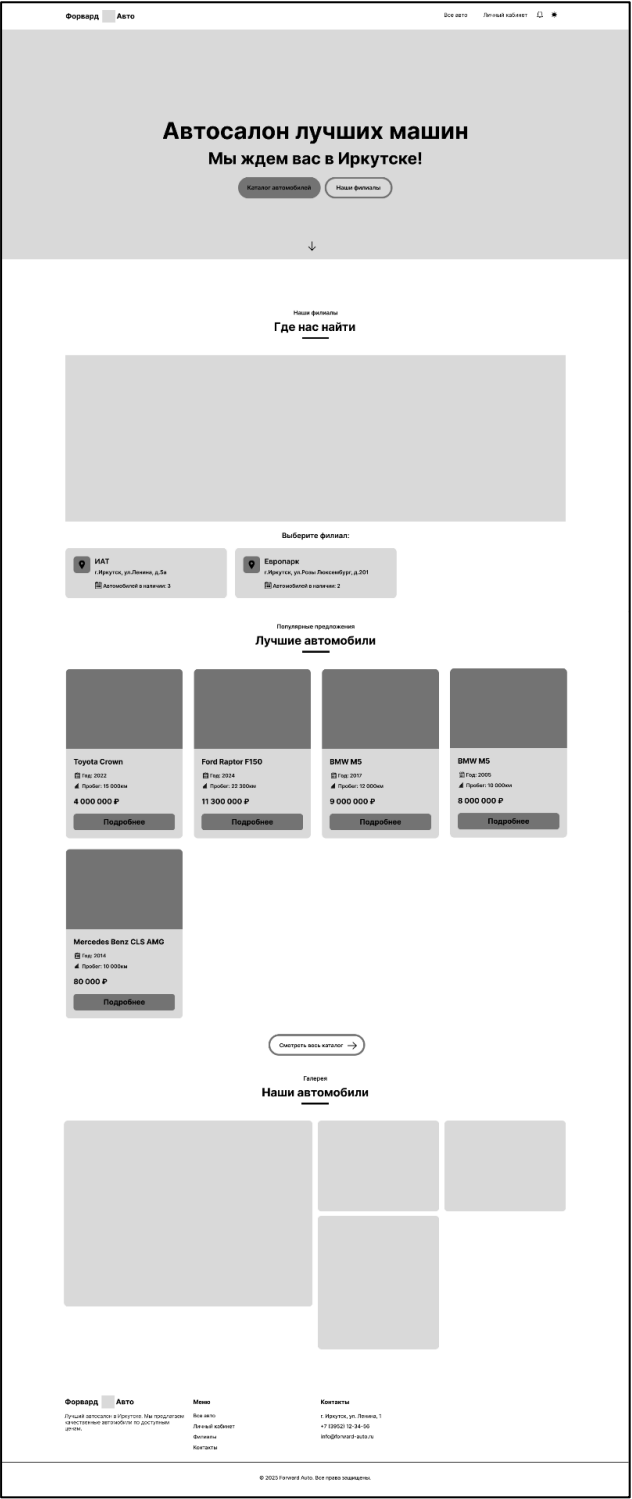


Рисунок 5 – Прототип главной страницы

На главной странице пользователь может, найти необходимую услугу, перейти на страницу каталога со всеми авто, посмотреть адреса филиалов на карте, и зайти в личный кабинет.

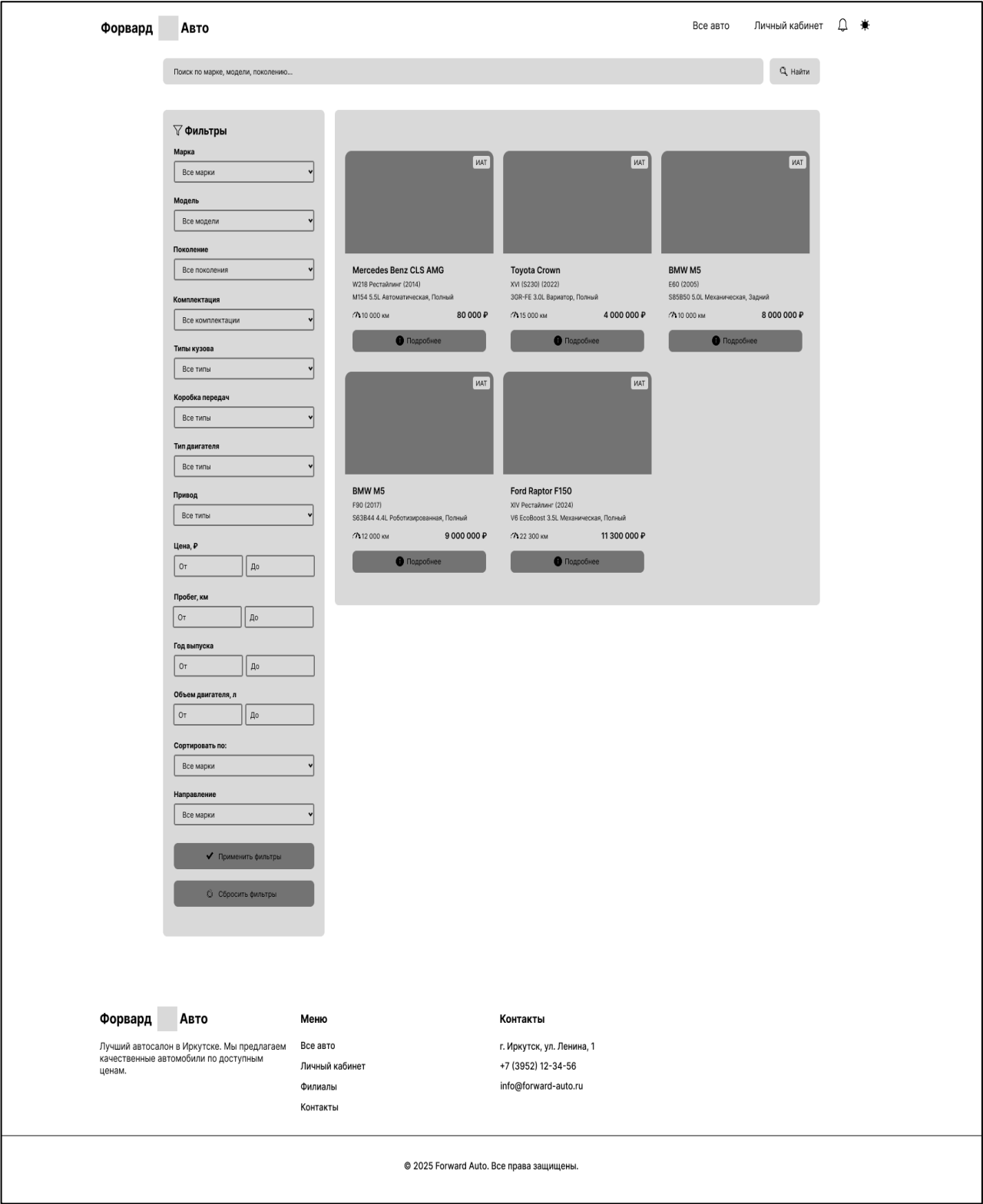


Рисунок 6 – Прототип страницы каталога

На данной странице пользователь может посмотреть какие авто сейчас находятся в продаже, найти интересующий авто и отсортировать по параметрам.

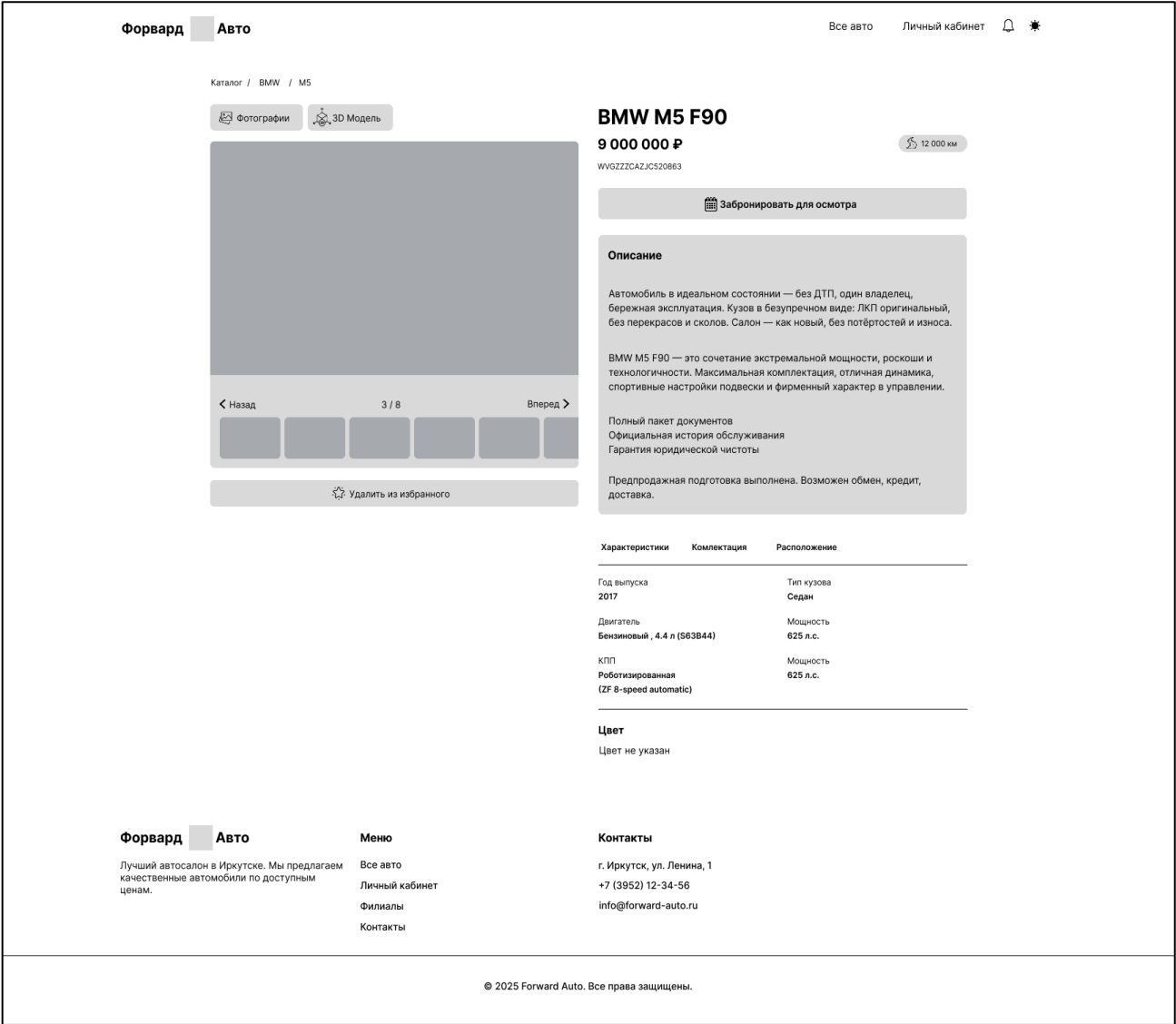


Рисунок 7 – Прототип страницы автомобиля

На этой странице пользователь может подробно ознакомиться с предложением о продаже автомобиля, изучить его технические характеристики, узнать о доступных вариантах комплектации, просмотреть фотографии и дополнительное описание. Также на странице предусмотрены удобные инструменты для добавления автомобиля в избранное, получения более подробной консультации у продавца, а также для оформления заявки на покупку с дальнейшим взаимодействием в рамках процесса приобретения авто.

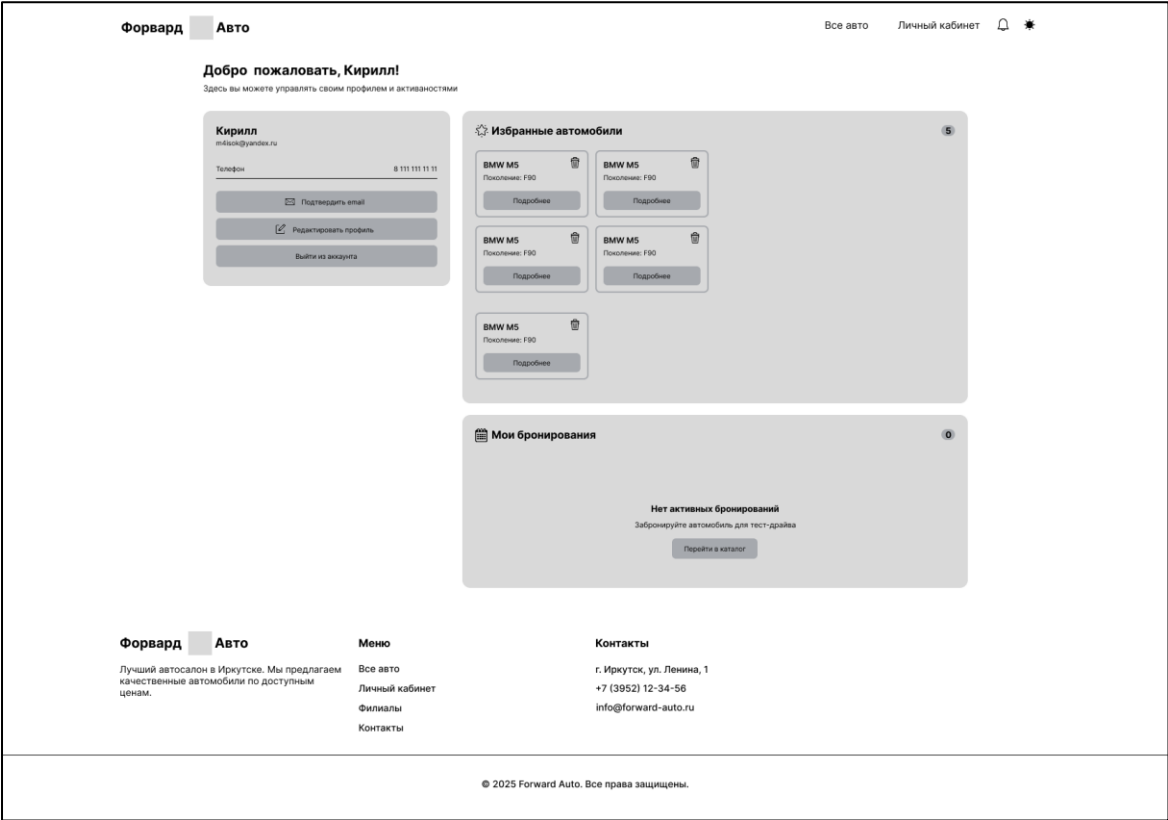


Рисунок 8 – Прототип страницы личного кабинета

На данной странице пользователь может поменять личные данные, ознакомиться со списком бронирований и списком «избранных» автомобилей.



Рисунок 9 – Оформление заявки

На представленной странице пользователь может полностью посмотреть список «избранных» автомобилей, также может удалить авто из «избранного» или перейти на страницу продажи самого автомобиля.

3.4.2 Выбор цветовой гаммы и шрифтов

Для создания единого стиля интерфейса веб-приложения были подобраны цвета и шрифты, обеспечивающие комфортный опыт взаимодействия.

Веб-приложение выполнено в сочетании фиолетово-белого цвета, для достижения единой цветовой гармонии (Рисунок 10).

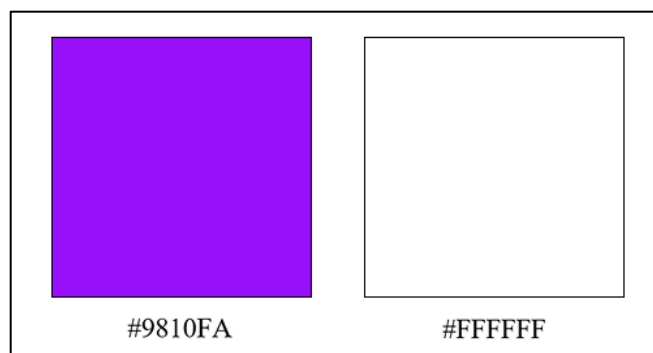


Рисунок 10 – Цветовая гамма

Основным шрифтом для веб-приложения был выбран «Instrument Sans», который обеспечивает хорошую читаемость заголовков и основного текста.

3.4.3 Разработка элементов интерфейса

Для наиболее удобного использования веб приложения автосалона были разработаны такие элементы как: поле ввода, кнопка, карточка услуги.

Поле ввода – это графический элемент, предназначенный для ввода текста. В веб-приложении оно имеет серый фон, с черной обводкой (рисунок 11).

example@gmail.com

Рисунок 11 – Поле ввода информационной системы

Кнопка – важный элемент, с помощью которого совершаются какое-либо действие. В веб-приложении кнопка выполнена в фиолетовом цвете с белым текстом внутри (рисунок 12).

Войти

Рисунок 12 – Кнопка веб-приложения

Карточка автомобиля – элемент интерфейса, благодаря которому, пользователь может кратко ознакомиться с основной информацией об автомобиле на странице каталога (рисунок 13).

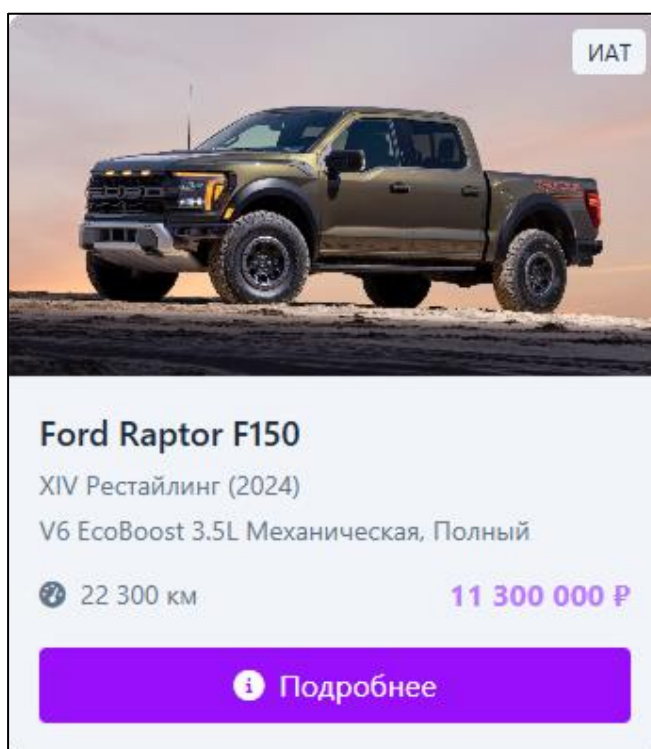


Рисунок 13 – Карточка автомобиля

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						29
Изм.	Лист	№ докум.	Подпись	Дата		

3.4.4 Разработка дизайн макетов

После проектирования ранее упомянутых элементов была начата разработка дизайн-макета веб-приложения.

Далее на рисунках 13 – 17 представлены дизайн-макеты страниц

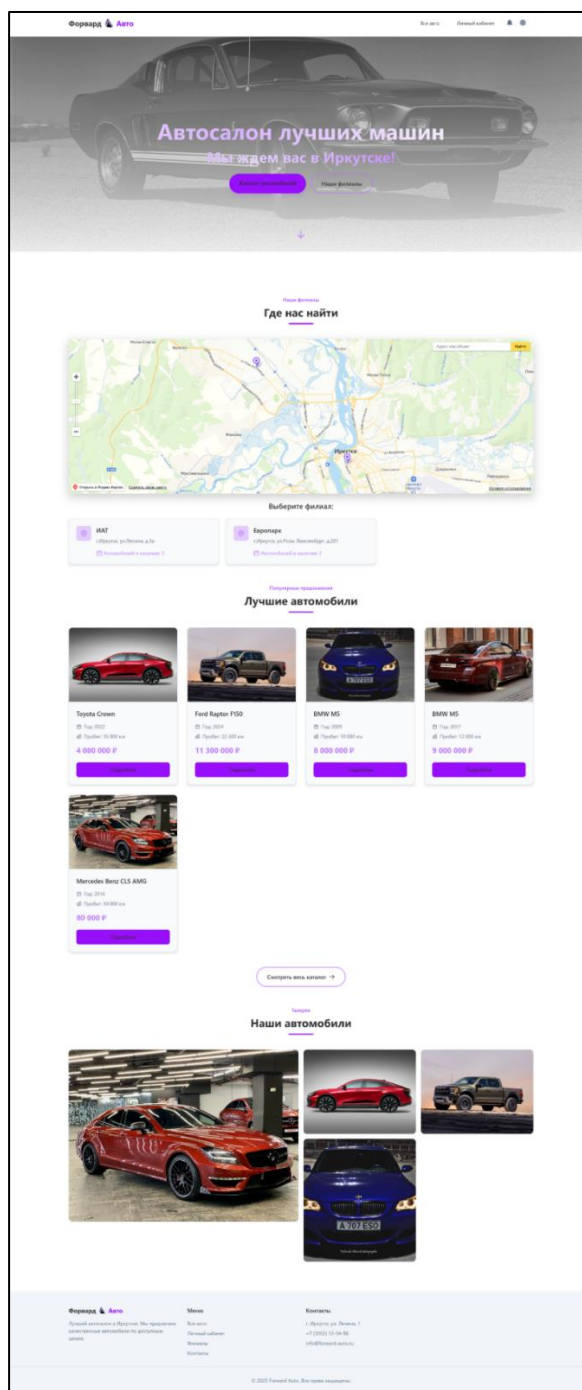


Рисунок 13 – Дизайн-макет главной страницы

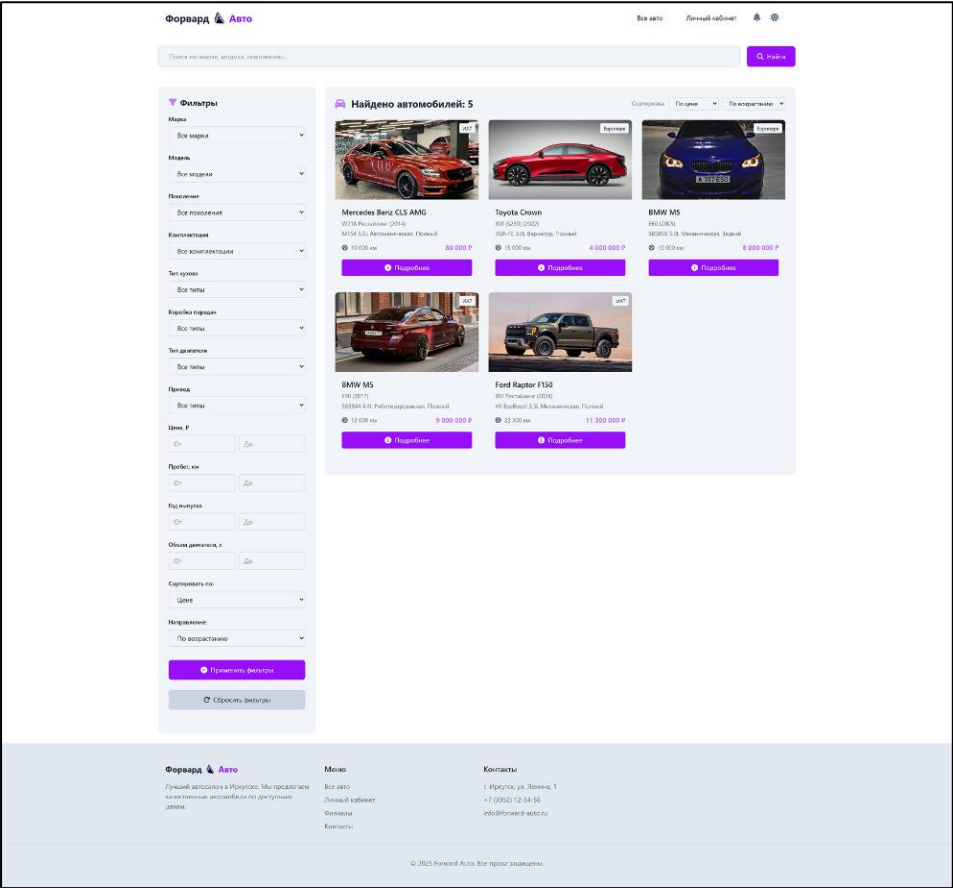


Рисунок 14 – Каталог автомобилей

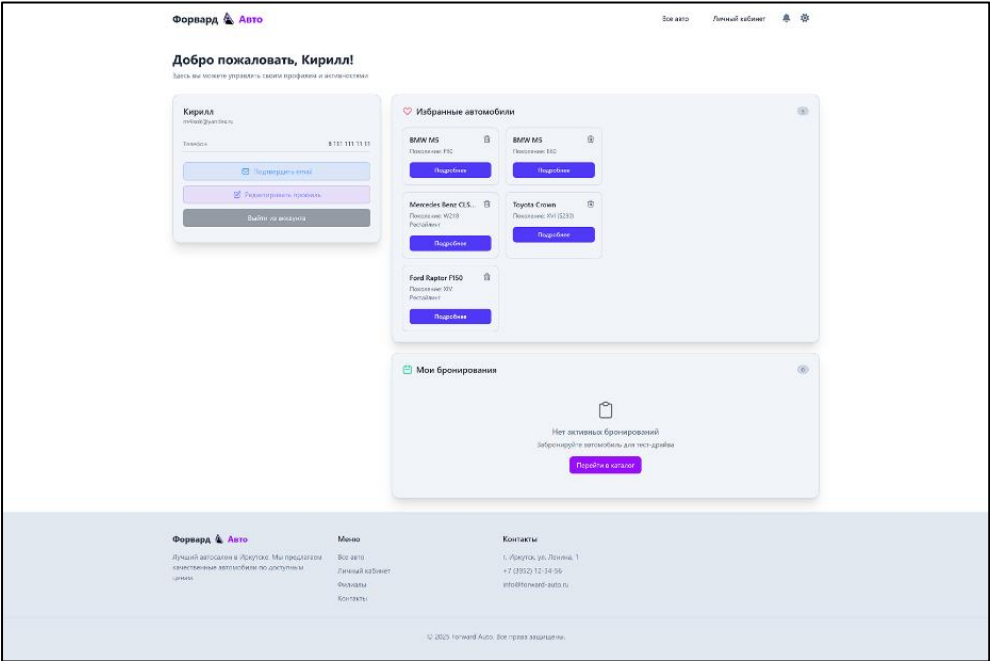


Рисунок 15 – Личный кабинет

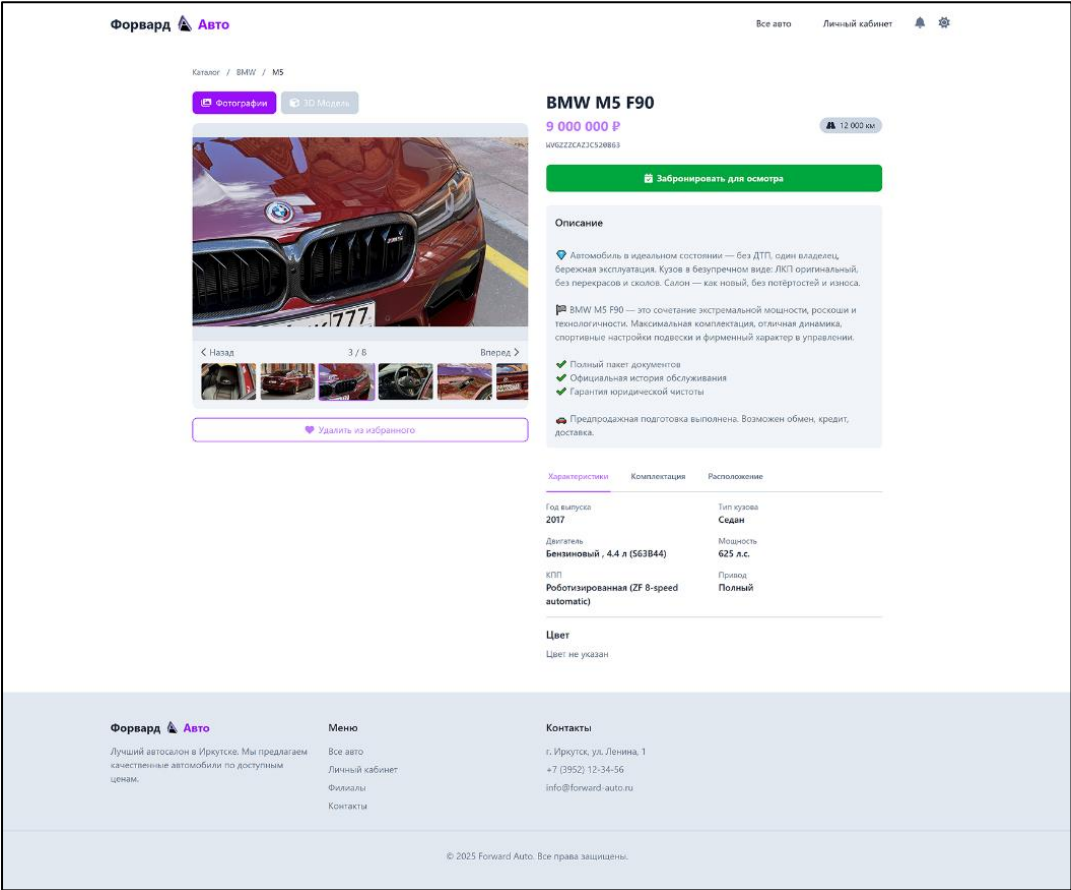


Рисунок 16 – Страница авто

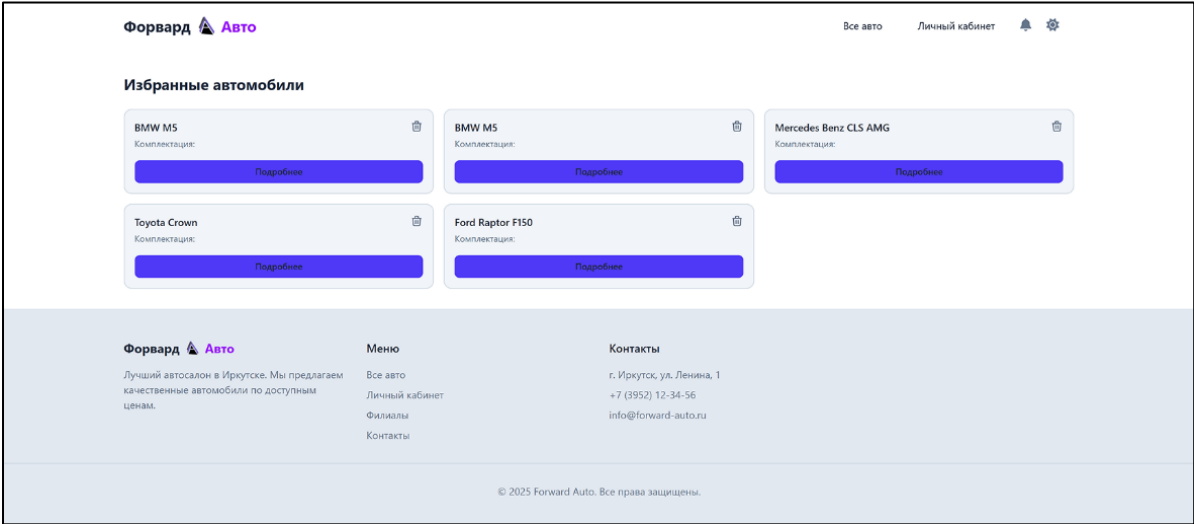


Рисунок 17 – Страница избранного

4 Реализация программного продукта

4.1 Разработка клиентской части

Разработка клиентской части веб-приложения является важным этапом, ведь это определяет пользовательский опыт и так же может влиять на производительность и безопасность. В дипломном проекте клиентская часть реализована с помощью фреймворка Tailwind. На рисунке 18 представлен код макет всех клиентских страниц (user.blade.php).

```
<div class="min-h-screen">
  <header class="main-header">
    <div class="container mx-auto px-4 py-4 flex justify-between items-center">
      <a href="{{route('welcome')}}" class="flex items-center space-x-2">
        <span class="text-2xl font-bold">
          Форвард
        </span>
        
        <span class="text-2xl font-bold text-purple-600">
          Авто
        </span>
      </a>

      <div class="flex items-center space-x-6">
        <nav id="menu" class="hidden md:flex md:space-x-6 items-center">
          <a class="px-4 py-2 rounded-lg hover:bg-purple-600/20 transition-all" href="{{route('catalog')}}">Все авто</a>
          <a class="px-4 py-2 rounded-lg hover:bg-purple-600/20 transition-all" href="{{route('dashboard')}}">Личный кабинет</a>
          @auth
            @if(auth()->user()->isManager())
              <a class="px-4 py-2 rounded-lg hover:bg-purple-600/20 transition-all" href="{{route('manager.bookings.index')}}">Менеджер панель</a>
            @endif
            @if(auth()->user()->isAdmin())
              <a class="px-4 py-2 rounded-lg hover:bg-purple-600/20 transition-all" href="{{route('admin.car-structure.index')}}">Админ панель</a>
            @endif
          @endauth
        </nav>

        @auth
          <div class="notification-icon relative">
            <a href="#" id="notification-bell" class="text-gray-300 hover:text-white relative">
              <i class="fas fa-bell text-xl"></i>
              @if(auth()->user()->unreadNotifications()->count() > 0)
                <span class="notification-badge absolute -top-1 -right-1 bg-red-500 text-white rounded-full flex items-center justify-center">
                  {{ auth()->user()->unreadNotifications()->count() }}
                </span>
              @endif
            </a>

            <div class="notification-dropdown hidden absolute mt-2 bg-[#2A2A2A] rounded-md shadow-lg z-50 border border-gray-700">
              <div class="p-3 bg-[#2A2A2A]">
                <div class="flex justify-between items-center">
                  <h4 class="font-semibold">Уведомления</h4>
                  <a href="#" id="mark-all-read" class="text-xs text-purple-400 hover:text-purple-300">Отметить все как прочитанные</a>
                </div>
              </div>
            </div>
          @endauth
        </div>
      </div>
    </div>
  </div>
```

Рисунок 18 – Макет страниц

После файл макета страниц подключается на всех клиентских страницах, определяя базовую структуру всех HTML-страниц веб-приложения. На рисунке 19 представлен код страницы каталога с использованием макета.

```

@extends('layouts.user')

@section('content')
<!-- Глобальный поиск -->
<section class="p-6 bg-[#1C1B21]">
  <div class="container mx-auto">
    <form action="{{ route('catalog') }}" method="GET" class="flex flex-col md:flex-row gap-4">
      <input type="text" name="search" value="{{ request('search') }}" placeholder="Поиск по марке, модели, поколению..."
        class="flex-1 px-6 py-3 rounded-lg border border-gray-700 bg-[#2A2A2A] text-white focus:outline-none
        focus:ring-2 focus:ring-purple-500 transition">
      <button type="submit" class="px-6 py-3 bg-purple-600 hover:bg-purple-700 rounded-lg text-[#ffffff]
        transition flex items-center justify-center">
        <i class="fas fa-search mr-2"></i> Найти
      </button>
      <if (request('search'))>
        <a href="{{ route('catalog') }}" class="px-6 py-3 bg-gray-700 hover:bg-gray-600 rounded-lg transition
        flex items-center justify-center">
          <i class="fas fa-times mr-2"></i> Сбросить
        </a>
      </if>
    </form>
  </div>
</section>

<section class="p-6 bg-[#1C1B21]">
  <div class="container mx-auto">
    <div class="flex flex-col lg:flex-row gap-6">
      <!-- Фильтры -->
      <div class="lg:w-1/4">
        <div class="filter-sidebar bg-[#2A2A2A] rounded-lg p-6">
          <h3 class="text-xl font-bold mb-4 flex items-center">
            <i class="fas fa-filter mr-2 text-purple-400"></i> Фильтры
          </h3>

          <form method="GET" action="{{ route('catalog') }}">
            <input type="hidden" name="search" value="{{ request('search') }}">

            <div class="space-y-6">
              <!-- Карта -->
              <div>
                <label class="block text-sm font-medium mb-2">Карта</label>
                <select name="brand" class="w-full px-4 py-2 rounded-md border border-gray-700 bg-[#3C3C3C]
                  text-white focus:ring-2 focus:ring-purple-500 focus:border-transparent transition">
                  <option value="">Все марки</option>
                  @foreach($brands as $brand)
                    <option value="{{ $brand->id }}" {{ request('brand') == $brand->id ? 'selected' : '' }}>
                      {{ $brand->name }}
                    </option>
                  @endforeach
                </select>
              </div>
            </div>
          </form>
        </div>
      </div>
    </div>
  </div>
</section>

```

Рисунок 19 – Страница каталога

Далее шаблон страницы администратора использует структуру другого макета (admin.blade.php) с помощью директивы «@extends('layouts.admin)». Это позволяет разделять макетами страницы админа и клиента. Основной контент страницы админа определяется в секции «@section('content')», которая затем вставляется в соответствующую область макета (@yield('content')). Часть кода шаблона администратора представлено на рисунке 20. С полным кодом макета администратора можно ознакомиться в приложении Б.

```

@extends('layouts.admin')

@section('title', 'Структура автомобилей')

@section('content')


<!-- Header -->
    <div class="mb-8">
        <h1 class="text-3xl font-bold text-gray-900">Управление автомобильной структурой</h1>
        <p class="mt-2 text-sm text-gray-600">Управление марками, моделями и поколениями автомобилей</p>
    </div>

    <!-- Messages -->
    @if(session('error'))
        <div class="bg-red-50 border-1-4 border-red-400 p-4 mb-6 rounded">
            <div class="flex">
                <div class="flex-shrink-0">
                    <svg class="h-5 w-5 text-red-400" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor">
                        <path fill-rule="evenodd" d="M10 18a8 8 0 10-16 8 8 8 0 1016 8z" />
                    </svg>
                </div>
                <div class="ml-3">
                    <p class="text-sm text-red-700">{{ session('error') }}</p>
                </div>
            </div>
        </div>
    @endif

    @if(session('success'))
        <div class="bg-green-50 border-1-4 border-green-400 p-4 mb-6 rounded">
            <div class="flex">
                <div class="flex-shrink-0">
                    <svg class="h-5 w-5 text-green-400" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor">
                        <path fill-rule="evenodd" d="M10 18a8 8 0 10-16 8 8 8 0 1016 8z" />
                    </svg>
                </div>
                <div class="ml-3">
                    <p class="text-sm text-green-700">{{ session('success') }}</p>
                </div>
            </div>
        </div>
    @endif

    @if ($errors->any())
        <div class="bg-red-50 border-1-4 border-red-400 p-4 mb-6 rounded">
            <div class="flex">
                <div class="flex-shrink-0">
                    <svg class="h-5 w-5 text-red-400" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor">
                        <path fill-rule="evenodd" d="M10 18a8 8 0 10-16 8 8 8 0 1016 8z" />
                    </svg>
                </div>
            </div>
        </div>
    @endif


```

Рисунок 20 – Часть страницы администратора

4.2 Разработка базы данных

База данных веб-приложения состоит из 18 таблиц. Таблицы были созданы с использованием миграций Laravel. Это упростило процесс управления базой данных в ходе разработки. Все файлы миграций представлены на рисунке 21.

```

❏ 0001_01_01_000000_create_users_table.php
❏ 0001_01_01_000001_create_cache_table.php
❏ 0001_01_01_000002_create_jobs_table.php
❏ 2025_05_02_202758_create_brands_table.php
❏ 2025_05_02_202758_create_countries_table.php
❏ 2025_05_02_202759_create_body_types_table.php
❏ 2025_05_02_202759_create_car_models_table.php
❏ 2025_05_02_202759_create_generations_table.php
❏ 2025_05_02_202800_create_drive_types_table.php
❏ 2025_05_02_202800_create_engine_types_table.php
❏ 2025_05_02_202800_create_transmission_types_table.php
❏ 2025_05_02_202801_create_colors_table.php
❏ 2025_05_02_205006_create_equipment_table.php
❏ 2025_05_02_205255_create_equipment_colors_table.php
❏ 2025_05_03_111900_create_branches_table.php
❏ 2025_05_05_122932_create_cars_table.php
❏ 2025_05_05_123009_create_car_images_table.php
❏ 2025_05_10_195533_create_bookings_table.php
❏ 2025_05_11_095756_create_favorite equipments_table.php
❏ 2025_05_11_095917_create_notifications_table.php

```

Рисунок 21 – Все файлы миграций

На рисунке 22 в качестве примера приведен код одного из файлов – создание таблицы авто.

```

Schema::create('cars', function (Blueprint $table) {
    $table->id();
    $table->foreignId('equipment_id')->constrained();
    $table->string('vin', 17)->unique();
    $table->integer('mileage');
    $table->decimal('price', 12, 2);
    $table->text('description')->nullable();
    $table->boolean('is_sold')->default(false);
    $table->foreignId('branch_id')->nullable()->constrained()->onDelete('set null');
    $table->foreignId('color_id')->nullable()->constrained()->onDelete('set null');
    $table->string('custom_color_name', 50)->nullable();
    $table->string('custom_color_hex', 7)->nullable();
    $table->timestamps();
});

```

Рисунок 22 – Миграция таблицы авто

4.3 Разработка серверной части

Первым и самым важным этапом в разработке серверной части веб-приложения, является создание моделей, которые позволяют упростить работу с запросами к базе данных. На рисунке 23 представлена часть кода модели авто (Car).

```
class Car extends Model
{
    protected $fillable = [
        'equipment_id', 'vin', 'mileage', 'price', 'description', 'is_sold', 'branch_id',
        'color_id', 'custom_color_name', 'custom_color_hex'
    ];

    public function equipment()
    {
        return $this->belongsTo(Equipment::class);
    }

    protected $appends = ['is_booked_by_me'];

    public function getIsBookedByMeAttribute()
    {
        return $this->bookings()
            ->where('user_id', Auth::id())
            ->whereIn('status', ['pending', 'confirmed'])
            ->exists();
    }

    public function images()
    {
        return $this->hasMany(CarImage::class);
    }

    public function getMainImageAttribute()
    {
        return $this->images->where('is_main', true)->first() ?? $this->images->first();
    }

    public function branch()
    {
        return $this->belongsTo(Branch::class);
    }
}
```

Рисунок 23 – Модель авто

После создания необходимых моделей, необходимо создать контроллеры, которые будут обрабатывать все HTTP-запросы. Часть кода контроллера каталога представлено на рисунке 24. Полный код контроллера CatalogController представлен в приложении В.

```

class CatalogController extends Controller
{
    public function index(Request $request)
    {
        // Базовый запрос
        $query = Car::with([
            'equipment.generation.carModel.brand',
            'equipment.bodyType',
            'equipment.engineType',
            'equipment.transmissionType',
            'equipment.driveType'
        ])->where('is_sold', false);

        // Глобальный поиск
        if ($request->search) {
            $query->where(function($q) use ($request) {
                $q->whereHas('equipment.generation.carModel.brand', function($q) use ($request) {
                    $q->where('name', 'like', "%{$request->search}%");
                })->orWhereHas('equipment.generation.carModel', function($q) use ($request) {
                    $q->where('name', 'like', "%{$request->search}%");
                })->orWhereHas('equipment.generation', function($q) use ($request) {
                    $q->where('name', 'like', "%{$request->search}%");
                });
            });
        }

        // Фильтрация по марке
        if ($request->brand) {
            $query->whereHas('equipment.generation.carModel.brand', function($q) use ($request) {
                $q->where('id', $request->brand);
            });
        }

        // Фильтрация по модели
        if ($request->model) {
            $query->whereHas('equipment.generation.carModel', function($q) use ($request) {
                $q->where('id', $request->model);
            });
        }

        // Фильтрация по поколению
        if ($request->generation) {
            $query->whereHas('equipment.generation', function($q) use ($request) {
                $q->where('id', $request->generation);
            });
        }

        // Фильтрация по комплектации
        if ($request->equipment) {
            $query->where('equipment_id', $request->equipment);
        }
    }
}

```

Рисунок 24 – Часть контроллера комплектаций

5 Отладка и тестирование программного продукта

Одним из этапов разработки любого программного продукта является тестирование и отладка.

Тестирование – это процесс выполнения программного продукта с целью выявления ошибок, несоответствий требованиям и оценки его качества. Главная цель тестирования веб-приложения – обеспечить надежность, безопасность и эффективность функционирования.

Отладка – это процесс поиска и устранения ошибок в программном коде. Отладка проводилась в процессе написания программного кода, методом пошагового выполнения кода.

Веб-приложение тестировалось с помощью PHPUnit, включая следующие методы:

- Интеграционное тестирование – проверка взаимодействия контроллеров, базы данных и HTTP-запросов.
- Функциональное тестирование – валидация (создание, обновление, удаление сотрудников).
- Тестирование безопасности – проверка роли пользователя.

Для тестирования веб-приложения был разработан сценарий тестирования для добавления комплектации. В таблице 12 представлен данный сценарий.

Таблица 19 – Сценарий тестирования добавления комплектации.

Поле	Описание
Дата тестирования	04.05.2025
Приоритет тестирования	Высокий

Продолжение таблицы 19

Поле	Описание
Этапы теста	<ol style="list-style-type: none"> 1. Авторизация в системе с правами администратора 2. Создание комплектации с корректными данными 3. Попытка создания комплектации без прав администратора 4. Проверка валидации обязательных полей 5. Проверка валидации числовых полей
Тестовые данные	Данные для создания комплектации: generation_id - ID поколения (XV70), body_type_id - ID типа кузова (Sedan), engine_type_id - ID типа двигателя (Gasoline) engine_power – 301, transmission_type_id - ID типа трансмиссии (Automatic), drive_type_id - ID типа привода (Front-wheel drive), country_id - ID страны (Japan) , description - Top of the line, range – 500 Некорректные данные: generation_id - "not-a-number" engine_volume - "not-a-number" engine_power - "invalid"
Ожидаемый результат	<ol style="list-style-type: none"> 1. Система создает новую комплектацию для администратора 2. Система запрещает создание для не-администратора (403 или редирект) 3. Система возвращает ошибки валидации для обязательных полей 4. Система возвращает ошибки валидации для числовых полей
Фактический результат	Все ожидаемые результаты соответствуют фактическим:

На рисунке 25, 26 представлены одна из функций тестирования и результаты всего теста.

```

public function test_non_admin_cannot_create_equipment()
{
    $user = User::factory()->create(['role' => 'manager']);

    $response = $this->actingAs($user)
        ->post(route('admin.equipments.store'), []);

    // Check for either 403 or redirect to home
    if ($response->status() !== 403) {
        $response->assertRedirect('/');
    } else {
        $response->assertStatus(403);
    }
}

```

Рисунок 25 – Функция тестирования


```

PASS Tests\Feature\EquipmentTest
✓ admin can create equipment
✓ non admin cannot create equipment
✓ required fields are validated
✓ numeric fields validation

```

Рисунок 26 – Результаты теста добавления комплектации

Далее был разработан второй сценарий тестирования, редактирование профиля администратором, менеджером и обычным пользователем.

Таблица 20 – Сценарий тестирования операций с инвентарем

Поле	Описание
Дата тестирования	04.05.2025
Приоритет тестирования	Высокий
Этапы теста	1. Редактирование профиля обычным пользователем 2. Проверка сброса верификации email при изменении 3. Попытка редактирования профиля администратором 4. Попытка редактирования профиля менеджером 5. Проверка обязательности поля имени 6. Проверка обязательности и уникальности email 7. Проверка обязательности, уникальности и формата телефона
Тестовые данные	Данные пользователя: Имя - Иван Иванов Email - user@example.com Телефон - 8 123 456 78 90 Роль - user Данные для обновления (корректные): Имя - Новое имя Email - new-user@example.com Телефон - 8 999 999 99 99 Некорректные данные: 1. Пустое имя 2. Пустой email 3. Занятый email (other@example.com) 4. Пустой телефон 5. Неверный формат телефона (1234567890) 6. Занятый телефон (8 999 999 99 99)

Продолжение таблицы 20

Поле	Описание
Ожидаемый результат	<ol style="list-style-type: none"> 1. Пользователь может обновить свои данные (имя, email, телефон) 2. При изменении email сбрасывается верификация 3. Администратор не может редактировать профиль (ошибка) 4. Менеджер не может редактировать профиль (ошибка) 5. Система требует указать имя 6. Система требует уникальный и валидный email 7. Система требует уникальный телефон в правильном формате
Фактический результат	Все ожидаемые результаты соответствуют фактическим:

На рисунке 27, 28 представлены одна из функций тестирования, а именно возможность редактировать профиль пользователем.

```

public function test_user_can_edit_profile()
{
    $user = User::factory()->create([
        'role' => 'user',
        'email' => 'user@example.com',
        'phone' => '8 123 456 78 90'
    ]);

    $updatedData = [
        'name' => 'Новое имя',
        'email' => 'new-user@example.com',
        'phone' => '8 999 999 99 99',
    ];

    $response = $this->actingAs($user)->put('/profile', $updatedData);

    $response->assertRedirect('/dashboard');
    $this->assertDatabaseHas('users', [
        'name' => 'Новое имя',
        'email' => 'new-user@example.com',
        'phone' => '8 999 999 99 99',
    ]);
    $this->assertEquals(session('status'), 'Email изменен. Пожалуйста, подтвердите новый email.');
```

Рисунок 27 – Одна из функций тестирования

```

PASS Tests\Feature\ProfileTest
✓ user can edit profile
✓ email verification is reset when email changed
✓ admin cannot edit profile
✓ manager cannot edit profile
✓ name is required
✓ email is required and must be unique
✓ phone is required and must be unique and valid format
```

Рисунок 28 – Результаты теста

6 Документация программного продукта

6.1 Руководство пользователя

Для запуска веб-приложения необходимо запустить приложение OpenServer и включить его, чтобы иконка флажка поменяла цвет на зеленый. Далее в терминале необходимо зайти в директорию веб приложения, если это еще не было сделано, введя команду «`cd ./`» и нажав клавишу Tab выбрать нужную директорию, затем ввести команду «`php artisan serve`». Открыть еще один терминал, также зайти в директорию информационной системы и ввести команду «`npm run dev`».

Далее необходимо зайти в браузер и перейти по адресу «`http://localhost:8000`». После загрузки пользователь увидит главную страницу (рисунок 29) информационной системы.

На главной странице пользователь видит приветственное сообщение с названием автосалона и местом его расположения. Вверху находятся две кнопки, одна из которых ведет к каталогу автомобилей, а другая позволяет быстро перейти к разделу филиалов. Ниже расположена секция с картой и адресами представительств автосалона. Пользователь может выбрать интересующий его филиал и посмотреть сколько автомобилей доступно в наличии. При нажатии на конкретный филиал откроется список автомобилей, которые в нем находятся что позволяет заранее определиться с посещением нужного места. Также есть возможность перейти ко всем филиалам для более детального ознакомления. Ниже можно найти секцию с популярными автомобилями, представленными в продаже. Каждый автомобиль сопровождается фотографией основными характеристиками и ценой. Нажав на кнопку подробнее, пользователь может ознакомиться с полной информацией об автомобиле. Если же хочется посмотреть больше вариантов, то предусмотрена кнопка, которая ведет к полному каталогу машин с возможностью фильтрации и подбора по параметрам. Еще ниже расположена галерея с фотографиями автомобилей что позволяет пользователю визуально ознакомиться с тем какие машины представлены в салоне. Общая структура страницы удобна для

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						43
Изм.	Лист	№ докум.	Подпись	Дата		

навигации и дает понимание, где находится нужная информация без лишних кликов.

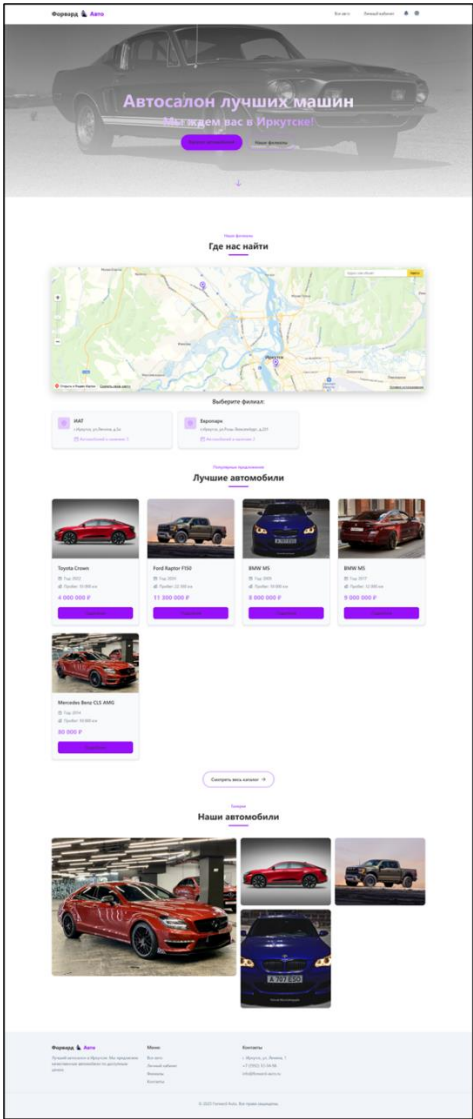


Рисунок 29 – Главная страница

На странице каталога (рисунок 30) пользователь видит поисковую строку, с помощью которой можно найти нужные автомобили по марке, модели или поколению. Ниже расположены фильтры, позволяющие уточнить выборку по различным параметрам: марка, модель, поколение, комплектация, тип кузова, коробка передач, тип двигателя, привод, цена, пробег, год выпуска и объем двигателя. Можно выбрать минимальные и максимальные значения для числовых параметров, а также задать направление сортировки. В правой части экрана

отображается список найденных автомобилей, каждый из которых имеет фото, основные характеристики и цену. Нажав на кнопку «Подробнее», пользователь перейдет на страницу с полной информацией об автомобиле. Если автомобилей не нашлось, будет предложено изменить параметры поиска или сбросить фильтры. Также в верхней части есть возможность быстро вернуться к начальным настройкам каталога.

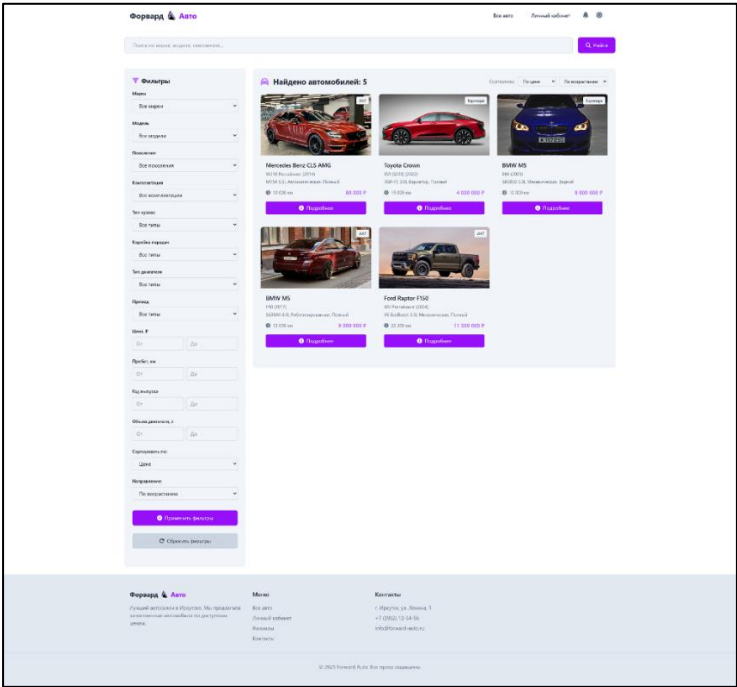


Рисунок 30 – Страница каталога

На странице автомобиля (рисунок 31) пользователь видит хлебные крошки, которые помогают понять, где он находится на сайте и быстро вернуться обратно в каталог или к выбранной марке. Ниже расположены уведомления о том, что автомобиль успешно забронирован или произошла ошибка это позволяет сразу получить обратную связь после действия. В левой части экрана находится основное изображение автомобиля с возможностью просмотра в модальном окне, а при наличии нескольких фото можно переключаться между ними с помощью кнопок или миниатюр. Также есть кнопка для просмотра 3D модели если она доступна. Для удобства пользователя реализована возможность добавить автомобиль в

избранное, которая сохраняет понравившиеся варианты для дальнейшего просмотра. В правой части страницы отображается подробная информация об автомобиле: марка модель и год выпуска пробег цена и VIN-номер если он указан. Ниже расположена кнопка для бронирования автомобиля на осмотр которая становится недоступной если пользователь уже авторизован, но не подтвердил email или если авто уже забронировано. Подробное описание автомобиля если оно заполнено поможет лучше ознакомиться с его состоянием и историей. Далее представлены вкладки с техническими характеристиками комплектацией и местоположением. На вкладке характеристик можно найти данные о двигателе трансмиссии приводе цвете и других параметрах. В разделе комплектации указаны дополнительные параметры такие как вес вместимость расход топлива и другие. На вкладке расположения показан адрес филиала, где находится автомобиль и карта для ориентира. Такая структура страницы позволяет пользователю получить всю необходимую информацию об автомобиле и принять решение о посещении салона.).

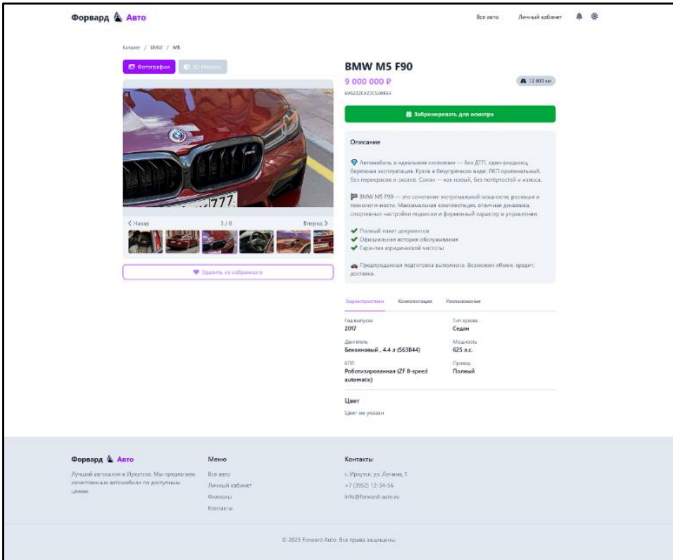


Рисунок 31- Страница авто

На странице регистрации (рисунок 32) пользователь видит форму с полями для ввода данных. Вверху отображается заголовок "Регистрация" и подсказка о

том, что если у пользователя уже есть аккаунт, он может войти, нажав на соответствующую ссылку. Форма содержит следующие поля: имя, email, телефон, пароль и подтверждение пароля. Для каждого поля предусмотрены сообщения об ошибках, которые появляются в случае некорректного ввода данных. Также указаны требования к паролю — минимум 8 символов, одна заглавная буква, одна строчная буква и цифра.

После заполнения всех полей пользователь может нажать кнопку "Зарегистрироваться", чтобы создать учетную запись. Если данные введены правильно, происходит переход на страницу авторизации или отображается сообщение об успешной регистрации. Если же возникают ошибки, они подсвечиваются рядом с соответствующими полями, и пользователь должен исправить их перед повторной отправкой формы.

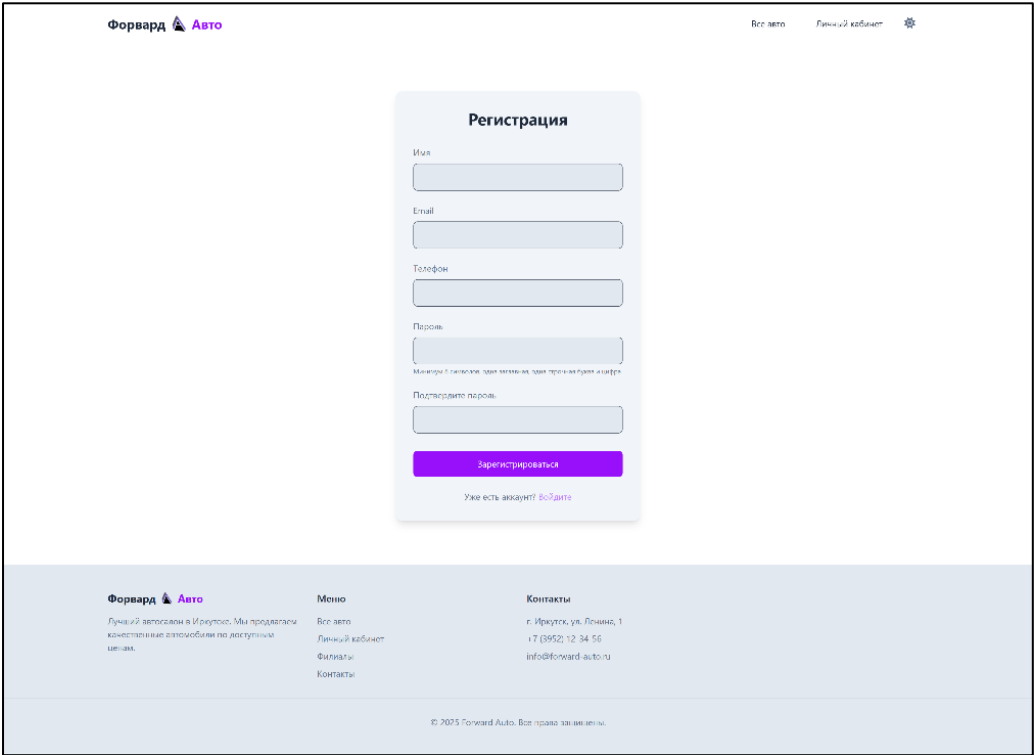


Рисунок 32 – Страница регистрации

Если пользователь уже зарегистрирован, то может перейти к странице входа, которая представлена на рисунке 33.

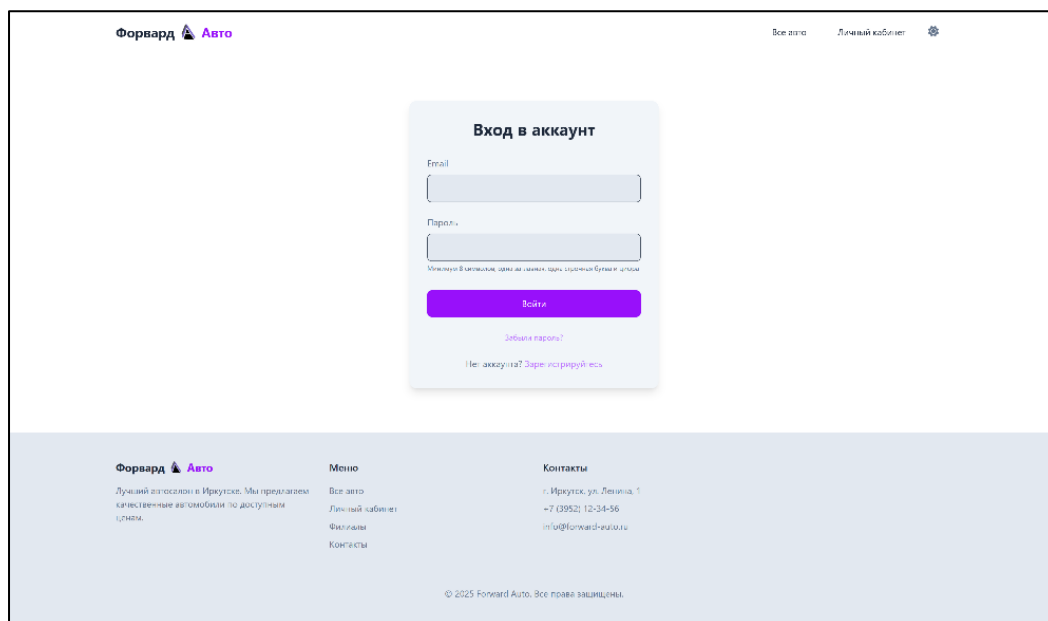


Рисунок 33 – Страница входа

На странице личного кабинета (рисунок 34) пользователь видит приветствие с указанием имени и основной информации о профиле. В левой части экрана находится блок с данными аккаунта: имя, email, телефон, статус (пользователь, менеджер или администратор). Также здесь доступны действия: подтверждение email, редактирование профиля и выход из аккаунта.

В правой части расположены два основных раздела. Первый — "Избранные автомобили", где отображаются до 10 машин, добавленных пользователем в избранное. Каждый автомобиль сопровождается названием и поколением. При необходимости можно перейти на страницу автомобиля или удалить его из избранного. Если автомобилей больше 10, есть ссылка для просмотра полного списка.

Второй раздел — "Мои бронирования". Здесь показаны все забронированные автомобили с датами визита и текущим статусом бронирования (ожидание, подтверждено, отклонено, отменено). Для каждого бронирования доступна кнопка "Подробнее", а также возможность отмены при наличии соответствующего статуса.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дата		

Пользователь может редактировать данные профиля или изменить пароль через модальное окно, которое открывается по кнопке "Редактировать профиль".

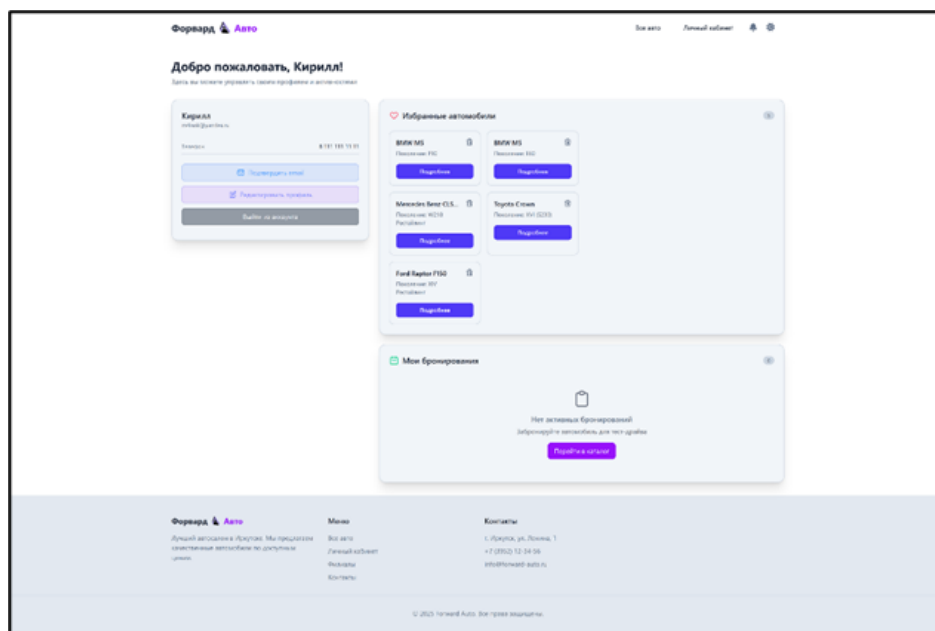


Рисунок 34 – Личный кабинет

6.2 Руководство администратора

На странице администратора (рисунок 35) пользователь видит панель управления, разделённую на блоки по типам данных: страны, филиалы, марки, модели, типы кузова, двигателя, привода и другие. Каждый блок содержит список существующих записей и форму добавления новой. Для каждой записи доступны действия: редактирование и удаление.

В верхней части есть строка поиска для фильтрации справочников. При вводе текста система автоматически подгружает подходящие результаты. Также можно сбросить фильтр, нажав на соответствующую кнопку.

Для добавления нового элемента, например, новой марки или типа двигателя, нужно заполнить поле ввода и нажать «Добавить». Чтобы изменить существующую запись, достаточно отредактировать текст в поле и нажать кнопку сохранения. Удаление выполняется через подтверждающее окно.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						49
Изм.	Лист	№ докум.	Подпись	Дата		

Все изменения сохраняются сразу, а при ошибках ввода система выводит уведомления рядом с полями.

Рисунок 35 – Управление референсами

На странице управления справочниками (рисунок 36) администратор видит список разделов: марки, модели, поколения и другие. Каждый раздел содержит таблицу с текущими записями и форму для добавления новых. В верхней части страницы есть строка поиска, которая фильтрует данные по названию.

Для добавления новой записи нужно заполнить поле ввода и нажать «Добавить». Чтобы отредактировать существующую запись, следует нажать кнопку редактирования, изменить данные и подтвердить сохранение. Удаление выполняется через модальное окно с подтверждением действия.

Все изменения применяются сразу, а при ошибках система выводит уведомления рядом с полями. Также доступна пагинация для просмотра других страниц справочника. Такая структура позволяет быстро управлять данными, не покидая интерфейс.

Референсы
Структура
Комплектации
Авто
Управление менеджерами
Управление бронированиями
Личный кабинет

Управление автомобильной структурой

Управление марками, моделями и поколениями автомобилей

Добавить марку
Добавить модель
Добавить поколение

Поиск и фильтрация

Марка

Выберите марку
x


Модель

Выберите модель


Поколение

Выберите поколение


Сбросить


BMW


▼
✎
🗑


Nissan


▼
✎
🗑


Toyota


▼
✎
🗑


Mercedes Benz

▼
✎
🗑


Cadillac

▼
✎
🗑


Porsche

▼
✎
🗑

Рисунок 36 – Управление структурой

На странице управления комплектациями (рисунок 38) администратор видит фильтры по марке и модели, форму поиска и кнопку добавления новой комплектации. Ниже отображается таблица с текущими записями. Каждая строка содержит информацию о марке, модели, поколении, типе двигателя, приводе, доступных цветах и действиях: редактирование и удаление.

В нижней части страницы находится раздел цветов. Здесь отображаются существующие цвета с их названием, НЕХ-кодом, визуальным обозначением и списком комплектаций, в которых они используются. Для каждого цвета доступны редактирование и удаление. Изменения сохраняются при нажатии на кнопку «Сохранить».

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						51
Изм.	Лист	№ докум.	Подпись	Дата		

Пагинация позволяет просматривать другие страницы списка. Сообщения об ошибках выводятся вверху, если данные введены некорректно или произошла ошибка при выполнении действия. (рисунок 37).

Комплектации

Фильтр комплектаций

Марка: Выберите значение | Модель: Выберите значение | Поколение: Выберите значение

Найти Сбросить

+ Добавить комплектацию

МАРКА	МОДЕЛЬ	ПОКОЛЕНИЕ	ДВИГАТЕЛЬ	ПРИВОД	ЦВЕТА	ДЕЙСТВИЯ
BMW	M5	F90	Бензиновый	Полный	Красный ашторин	
BMW	X7	G07	Бензиновый	Полный	Синий скатер рокс	
Nissan	CTR	R35	Бензиновый	Полный	Темно-синяя жемчужина	
Nissan	X-Trail	N32 II Рестайлинг	Бензиновый	Полный	Серебристый	
Toyota	Mark 2	JZX100	Бензиновый	Задний	Белый	
Toyota	Supra	X (J150)	Бензиновый	Полный	Белый	
Mercedes-Benz	C class	W205 Рестайлинг	Бензиновый	Полный	Белый	
Mercedes-Benz	S class	W222, C217 Рестайлинг	Бензиновый	Полный	Белый	
Cadillac	CTS	II	Бензиновый	Задний	Серебристый	
Cadillac	Escalade	V Рестайлинг	Бензиновый	Полный	Черный	

Showing 1 to 10 of 14 results

Управление цветами

Поиск цвета: Название или HEX код | Комплектация: Выберите значение | Найти | Сбросить

НАЗВАНИЕ	ЦВЕТ	HEX КОД	КОМПЛЕКТАЦИИ	СОХРАНИТЬ	УДАЛИТЬ
Красный ашторин		#C53777	BMW → M5 (F90) Porsche → Panamera (950/)		
Синий скатер рокс		#003251	BMW → X7 (G07) BMW → M5 (F90)		

Рисунок 37 – Управление комплектациями

На странице управления автомобилями (рисунок 38) администратор видит строку поиска, с помощью которой можно находить авто по VIN, марке, модели или поколению. Рядом расположены кнопки поиска и сброса фильтра. Ниже находится таблица со списком автомобилей, включающая информацию о VIN, марке/модели/поколении, цвете, цене, пробеге, филиале и статусе (на продаже или продан).

Каждая запись содержит две кнопки: «Редактировать» и «Удалить». Нажав на «Редактировать», администратор переходит к форме изменения данных автомобиля. Кнопка «Удалить» удаляет запись после подтверждения.

В правом верхнем углу есть кнопка «Добавить автомобиль», которая открывает форму для добавления нового автомобиля. Также доступна пагинация для навигации по страницам списка. Сообщения об ошибках или успешных действиях отображаются вверху, если такие имеются.

VIN	МАРКА / МОДЕЛЬ	ЦВЕТ	ЦЕНА	ПРОБЕГ	ФИЛИАЛ
WV78ZZC5545HJGHFG	Ford Raptor XIV	Красный	10 000 000 Р	20 000 км	Форвард Авто на Трактовой
WV23ZZC5545HJGHFG	Toyota Crown X (S150)	Белый	2 000 000 Р	750 000 км	Форвард Авто Интерлайн
WVGZZC5545HJGHFG	Mercedes Benz S class V1 (W222, C217) Рестайлинг	Белый	5 000 000 Р	25 000 км	Форвард Авто Интерлайн

Рисунок 38 – Управление авто

На странице управления менеджерами (рисунок 39) администратор видит форму добавления нового менеджера с полями: имя, email, телефон, пароль и подтверждение пароля. После заполнения всех полей можно нажать кнопку «Добавить менеджера», и новый пользователь появится в системе. Ниже отображается список уже добавленных менеджеров в виде таблицы с колонками: ID, имя, email, телефон и действия. Для каждого менеджера доступны две кнопки: редактировать — открывает возможность изменить данные (поля становятся редактируемыми), а также ввести новый пароль при необходимости. Кнопка "Сохранить" применяет изменения, удалить — удаляет запись после

подтверждения. После успешного добавления, редактирования или удаления выводится сообщение об успехе. Эта страница позволяет удобно управлять учетными записями менеджеров без лишних кликов.

Рисунок 39 – Управление менеджерами

6.3 Руководство менеджера

На странице управления бронированиями (рисунок 40) менеджер видит фильтры для поиска записей по статусу и дате визита. После выбора нужных параметров можно нажать «Применить», чтобы отобразить подходящие бронирования, или «Сбросить» для отображения всех записей.

Ниже расположен список бронирований в виде таблицы. Каждая запись содержит информацию: ID, автомобиль (марка, модель, VIN), клиента (имя, email, телефон), дату бронирования, дату визита и текущий статус. Статусы окрашены в разные цвета для удобного восприятия: ожидание, подтверждено, отклонено, завершено.

Для каждой записи доступны две кнопки:

Редактировать — открывает форму изменения статуса и других данных бронирования.

Удалить — удаляет запись после подтверждения.

Если бронирований нет или фильтры не дают результатов, отображается соответствующее сообщение. Также доступна пагинация для просмотра других страниц списка. Сообщения об успехе или ошибках выводятся вверху страницы при наличии.

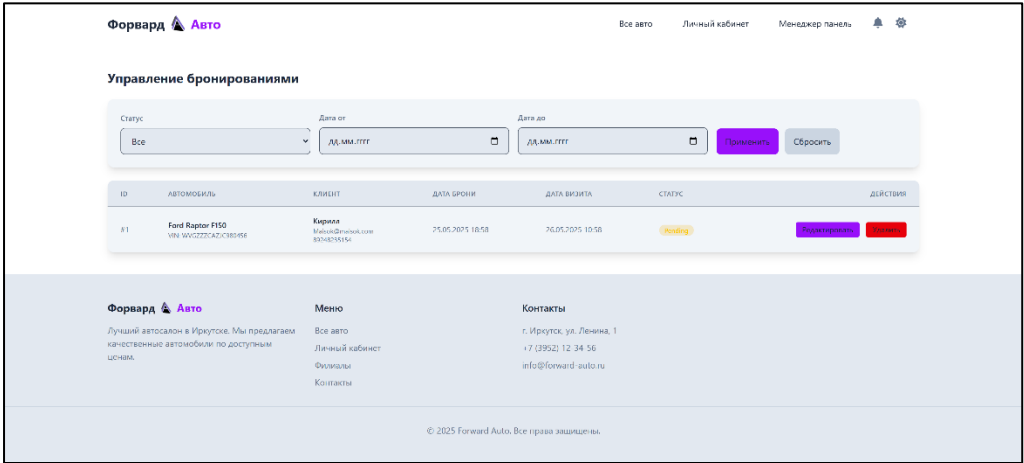


Рисунок 40 – Управление бронированиями

7 Стоимость разработки

Разработка веб-приложения для автосалона — это комплексный и многоуровневый процесс, требующий не только технической проработки, но и тщательного экономического обоснования. В условиях растущего спроса на цифровые сервисы и автоматизацию бизнес-процессов внедрение онлайн-платформы позволяет автосалонам выйти на новый уровень: повысить качество клиентского сервиса, упростить взаимодействие с покупателями и оптимизировать управление продажами автомобилей.

7.1 Расчет трудоемкости разработки программного обеспечения

Трудоемкость считается ключевым показателем экономической эффективности (характеристика расхода труда). Позволяет оценить использование рабочего времени в результате производственной деятельности или при оказании услуг. Формула трудоемкости показывает количество труда, затрачиваемое на единицу продукции.

Суммарные затраты труда рассчитываются как сумма составных затрат труда по формуле (1):

$$\sum T = t_{\text{оп}} + t_{\text{ан}} + t_{\text{пр}} + t_{\text{тз}} + t_{\text{разб}} + t_{\text{т}} + t_{\text{д}} \quad (1)$$

Таблица 21 – Суммарные затраты времени

Вид работ	Часы (всего)	Машинное время
Анализ предметной области проекта и описание структуры	20	20
Анализ инструментов	20	20
Проектирование проекта и описание этапов проектирования	50	50

Продолжение таблицы 21

Вид работ	Часы (всего)	Машинное время
Создание технического задания на разработку проекта	20	15
Разработка программного продукта	120	120
Тестирование программного продукта	40	40
Подготовка документации	30	20
Итого:	300	285

$$\sum T = 20 + 20 + 50 + 20 + 120 + 40 + 30 = 300$$

Из них:

Машинное время: 285 часов (разработка, тестирование).

Ручная работа: 15 часов (анализ, проектирование).

7.2 Затраты на оплату

Эффективная разработка программного продукта требует значительных трудозатрат.

Затраты на оплату (ЗОТ) труда разработчика ПО включают затраты на оплату труда и отчисления от фонда заработной платы.

Затраты на заработную плату определяются произведением часовой тарифной ставки разработчика и трудоемкости разработки программного продукта по следующей формуле:

Основная заработная плата рассчитывается в рублях по формуле (2)

$$З_{\text{осн}} = \sum t * TC_{\text{мес}} \quad (2)$$

Где $\sum t$ – суммарные затраты труда, вычисляемые по формуле (1), час.;

$TC_{\text{мес}}$ – часовая тарифная ставка, руб.

Часовая тарифная ставка: 800 руб.

Основная зарплата:

$$З_{\text{осн}} = 300 * 800 = 240\,000 \text{ руб.}$$

Разработчик является самозанятым, расчет налогового вычета будет от всей стоимости разработки

7.3 Затраты на амортизацию оборудования

Использование оборудования в процессе разработки приводит к его постепенному износу. Далее рассчитана сумма амортизационных отчислений для ноутбука и принтера, которые применялись при создании веб-приложения, что отражает часть затрат проекта.

Срок полезного использования согласно классификатору основных средств равен 2-3 года.

Для разработки использовался персональный компьютер стоимостью 106 674 руб. с учетом периферийного оборудования

Расчет амортизации производится при помощи данных формул (3-6):

$$A_n = \frac{100\%}{K_k} = \frac{100\%}{3} = 33,3\% \quad (3)$$

где

A_n – годовая норма амортизации;

K_k – срок полезного использования в соответствии с классификатором.

$$A_r = C_o * A_n = 106\,674 * 33,3\% = 35\,522 \quad (4)$$

где

A_r – ежегодная сумма амортизации;

C_o – начальная стоимость оборудования.

$$A_m = \frac{A_r}{12 \text{ мес.}} = \frac{35\,522}{12} = 2960 \quad (5)$$

где

A_m – ежемесячная сумма амортизации.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						58
Изм.	Лист	№ докум.	Подпись	Дата		

Далее необходимо сумму ежемесячной амортизации умножить на количество месяцев эксплуатации оборудования.

$$З_{\text{амор}} = A_{\text{м}} * m = 2960 * 1,7 = 5032 \quad (6)$$

Из таблицы 1 указано, что машинное время работы на компьютере 285 часов что равно 1,8 месяца.

Стоимость амортизации компьютера равна 5032 рублей.

7.4 Расчет затрат на электроэнергию

Эксплуатация технических средств сопровождается потреблением электроэнергии. Здесь приведен расчет затрат на электроэнергию, необходимую для работы ноутбука в течение всего периода разработки, что является важной составляющей общих расходов.

Для расчета затрат на электроэнергию первоочередное необходимо рассчитать расход электроэнергии оборудования по следующей формуле (7):

$$E = P * t \quad (7)$$

Где:

P – электрическая мощность в киловаттах;

t – время в часах.

Мощность компьютера: 0,5 кВт·ч.

$$E = 0,5 * 285 = 142,5$$

Согласно таблице 1, машинное время работы ноутбука равно 285 часов. Стоимость одного киловатт-часа равна 1,58 рубля для городского населения Иркутской области. Общее потребление равно 142,5 кВт·ч.

Расчет стоимости электроэнергии по следующей формуле (8):

$$C_{\text{э}} = E * Ц \quad (8)$$

где Ц – стоимость киловатт-час.

Затраты на электроэнергию:

$$C_{\text{э}} = 142,5 * 1,58 = 225,15 \text{ руб.}$$

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						59
Изм.	Лист	№ докум.	Подпись	Дата		

7.5 Затраты на материалы, израсходованные при проведении разработки и прочие

Для успешной реализации проекта требуются различные вспомогательные материалы. В данном разделе перечислены и рассчитаны затраты на канцелярские товары и другие расходные материалы, которые были использованы в ходе работы.

Таблица 22 – Затраты на материалы

Наименование	Цена за единицу (руб.)	Кол-во (шт.)	Всего (руб.)
Бумага (500 листов)	360	100	72
Сшивание диплома	550	1	550
Итого			622

7.6 Расчет затрат на разработку программного обеспечения

Затраты на разработку программного обеспечения включают в себя:

- 1) затраты на зарплату разработчика ($Z_{зп}$);
- 2) затраты на амортизацию оборудования ($Z_{аморт}$);
- 3) затраты на эксплуатацию оборудования (электроэнергия) ($Z_{экспл}$);
- 4) затраты на материалы, израсходованные при проведении разработки (бумага, картридж, и т.п.) ($Z_{мат}$);

Затраты на разработку рассчитываются путем суммы всех затрат по формуле 10.

$$Z_{разр} = Z_{зп} + Z_{аморт} + Z_{экспл} + Z_{мат} \quad (9)$$

Таблица 23 – Общие расходы

Статьи затрат	Индекс	Сумма, руб.	Удельный вес затрат, %
Заработная плата	Зп.	240 000	97,61
Амортизация оборудования	Заморт.	5032	2,05
Затраты на электроэнергию	Зэкспл.	225,15	0,09
Затраты на материалы	Змат.	622	0,25
Итого:	Зразр.	245 879,65	100

Самозанятые разработчики имеют право осуществлять договорные отношения с физическими лицами ставка, по которой рассчитывается налог, – 4% ($H_{фз}$)

Следовательно, данный вид налога рассчитывается от суммы разработки полностью по формуле 11.

$$H_{сз} = \frac{245\,879,65}{100\%} * 4\% = 9\,835,00 \quad (10)$$

7.7 Расчет цены

Определим расчётную цену при предполагаемом (плановом) размере прибыли на уровне 20%.

$$Pr = 245\,839,65 \times 20\% = 49\,175,93 \text{ руб.}$$

$$HДС = 0,2 * (245\,839,65 + 49\,167,93) = 59\,011,51 \text{ руб.}$$

Итоговая цена:

$$Ц = 245\,839,65 + 49\,167,93 + 59\,001,51 = 354\,067,09 \text{ руб.}$$

Итоговая стоимость программного продукта составляет 354 067,09 рублей.

ЗАКЛЮЧЕНИЕ

В рамках выполнения дипломного проекта была сформулирована основная задача – создание веб-приложения «автосалон». Для достижения этой задачи был сформирован перечень необходимых действий.

Для реализации веб-приложения были выбраны следующие технологии: MySQL и Laravel.

Задачи были успешно выполнены, и веб-приложение было разработано в соответствии с техническим заданием. В ходе работы над дипломным проектом были получены и использованы новые знания о технологиях и средствах разработки. Было создано веб-приложение, которое позволяет оформлять бронирования, обладает удобным интерфейсом и простой навигацией по содержимому.

В результате дипломного проектирования была достигнута основная цель – создание веб-приложения «автосалон». Были изучены средства и выполнены поставленные задачи для достижения цели:

- произвести исследование деятельности автосалона;
- определить инструментальные средства проектирования и реализации;
- разработать техническое задание для веб-приложения;
- спроектировать базу данных;
- спроектировать интерфейс пользователя веб-приложения;
- реализовать веб-приложение;
- провести тестирование веб-приложения;
- составить программную документацию.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 css-tricks.com – Tailwind CSS Examples and Articles – URL: <https://css-tricks.com> (дата обращения: 03.05.2025). – Текст: электронный.
- 2 dev.mysql.com – MySQL Documentation – URL: <https://dev.mysql.com/doc> (дата обращения: 30.04.2025). – Текст: электронный.
- 3 dev.to/t/laravel – Laravel Community Discussions – URL: <https://dev.to/t/laravel> (дата обращения: 06.05.2025). – Текст: электронный.
- 4 github.com/laravel/laravel – Laravel GitHub Repository – URL: <https://github.com/laravel/laravel> (дата обращения: 02.05.2025). – Текст: электронный.
- 5 laracasts.com – Video Lessons on Laravel and PHP – URL: <https://laracasts.com> (дата обращения: 24.04.2025). – Текст: электронный.
- 6 laravel.com – Official Documentation – URL: <https://laravel.com/docs> (дата обращения: 22.04.2025). – Текст: электронный.
- 7 laravel.ru – Laravel Documentation in Russian – URL: <https://laravel.ru/docs> (дата обращения: 23.04.2025). – Текст: электронный.
- 8 laravel-news.com – Laravel News and Updates – URL: <https://laravel-news.com> (дата обращения: 04.05.2025). – Текст: электронный.
- 9 nova.laravel.com – Laravel Nova Admin Panel – URL: <https://nova.laravel.com> (дата обращения: 01.05.2025). – Текст: электронный.
- 10 [plugins.jetbrains.com](https://plugins.jetbrains.com/plugin/9869-laravel) – Laravel Plugin for PhpStorm Guide – URL: <https://plugins.jetbrains.com/plugin/9869-laravel> (дата обращения: 29.04.2025). – Текст: электронный.
- 11 sqlzoo.net – Interactive SQL Tutorial – URL: <https://sqlzoo.net> (дата обращения: 25.04.2025). – Текст: электронный.
- 12 stackoverflow.com/questions/tagged/laravel – Laravel Questions and Answers – URL: <https://stackoverflow.com/questions/tagged/laravel> (дата обращения: 05.05.2025). – Текст: электронный.

13 tailwindcss.com – Official Tailwind CSS Documentation – URL: <https://tailwindcss.com/docs> (дата обращения: 27.04.2025). – Текст: электронный.

14 tailwindcss.ru – Tailwind CSS Russian Language Documentation – URL: <https://tailwindcss.ru/docs> (дата обращения: 28.04.2025). – Текст: электронный.

15 w3schools.com/sql – SQL Tutorial – URL: <https://www.w3schools.com/sql> (дата обращения: 26.04.2025). – Текст: электронный.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						64
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А Техническое задание

Министерство образования Иркутской области

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

Техническое задание

ВЕБ-ПРИЛОЖЕНИЕ «АВТОСАЛОН»

Руководитель:

(подпись, дата)

(П.А. Стош)

Студент:

(подпись, дата)

(К.С. Мясников)

Иркутск, 2025

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						65
Изм.	Лист	№ докум.	Подпись	Дата		

1 Введение

1.1 Общие сведения

Документ представляет собой техническое задание на создание веб-приложения автосалона, предназначенного для организации работы веб-приложения автосалона.

1.2 Цели и задачи

Целью дипломного проекта является создание веб-приложения, которое позволит автоматизировать процессы, происходящие в автосалоне.

Задачи веб-приложения включают:

- Управление бронированиями.
- Управление базой автомобилей.
- Управление менеджерами.
- Управление продаваемыми автомобилями.
- Управление менеджерами на бронированиях.

2 Основания для разработки

2.1 Нормативные документы

Документ основывается на следующих нормативных документах:

- ГОСТ 34.602-2020 "Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы".
- ГОСТ Р 56477-2015 "Проектирование и внедрение информационных систем. Общие требования".

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дата		

2.2 Проектные документы

Проектные документы включают:

- Пояснительную записку.
- Руководство пользователя.
- Задание на дипломный проект.

3 Назначение системы

3.1 Общее описание

Веб-приложение автосалона – это веб-приложение, предназначенное для автоматизации и оптимизации процессов, связанных с бронированием автомобилей, обслуживанием клиентов и управление данными автосалона.

3.2 Преимущества и новизна

Веб-приложение будет предоставлять:

- Интуитивно понятный интерфейс для пользователей.
- Автоматизацию процессов, связанных с управлением автомобилями, бронированиями, менеджерами и взаимодействием их с клиентами.

4 Требования к системе

4.1 Функциональные требования

Общие функции системы

- Авторизация пользователей.
- Регистрация пользователей.
- Операции с отображением данных с возможностью сортировки по возрастанию и убывания по значениям: год, пробег, год.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						67
Изм.	Лист	№ докум.	Подпись	Дата		

– Функции фильтрации по полям: марке, модели, поколению, комплектации, типам кузова, коробкам передач, типу привода.

– Функции поиска данных по марке, модели, поколению.

– Адаптация под различные типы устройства.

– Разграничение прав доступа.

– Интеграция с Яндекс картами для отображения филиалов на карте.

Пользователь:

– Оформление и отмена бронирований

– Редактирование данных аккаунта

Менеджер:

– На странице управления бронированиями менеджер может: принять в работу бронирование, редактировать дату визита, статус бронирования, комментарий менеджера.

– На странице доступна фильтрация по значениям: статуса, типа бронирования, дата от, дата до.

Администратор:

– На странице справочники добавление, удаление, редактирование: типов кузова, стран, типов двигателя, типов коробок передач, филиалов

– На странице доступен поиск по типам кузова, типам двигателя, типам коробок передач, филиалов, стран.

– На странице автомобильной структуры: добавление, удаление, редактирование: марок авто, моделей авто, поколений авто, на странице доступна фильтрация по значениям: марки авто, модели авто, поколению авто.

– На странице комплектаций: добавление, удаление, редактирование: комплектаций авто, цветов авто, доступен поиск по значениям: названия цвета, hex кода цвета, доступна фильтрация по значениям: марок авто, моделей авто, поколений авто.

– На странице автомобилей: добавление, удаление, редактирование автомобилей, на странице доступна фильтрация по значениям: марок авто, модели

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						68
Изм.	Лист	№ докум.	Подпись	Дата		

авто, поколений авто, на странице доступен поиск по vin номеру, марке, модели, поколению, на странице доступна сортировка по возрастанию и убыванию по значениям: пробег, цена.

– Формирование отчета с функцией экспорта таблицы Excel продаж авто на основе данных: марка, модель, поколение, комплектация, дата от, дата до.

– На странице менеджеров: добавление, удаление, редактирование менеджеров, поиск менеджеров по значению: имя электронная почта, номер телефона

– На странице бронирований: изменения менеджера у бронирования, на странице доступна фильтрация на основе значений: менеджер, статус, дата от, дата до.

– Формирование отчета с функцией экспорта таблицы Excel данных бронирований на основе данных: менеджер, статус, дата от, дата до.

4.2 Технические требования

Производительность:

- Обработка до 50 задач одновременно.
- Время отклика системы не более 7 секунд при загрузке данных.

Надежность:

- Доступность системы не менее 99,5% в год.

Безопасность:

- Регистрация и мониторинг всех действий пользователей.
- Шифрование данных на уровне передачи хранения.

4.3 Эксплуатационные требования

- Удобство использования.
- Дружественный пользовательский интерфейс.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						69
Изм.	Лист	№ докум.	Подпись	Дата		

5 Требования к техническому обеспечению

5.1 Оборудование

- Сервер: Серверная платформа с процессором не менее 8 ядер, 16 ГБ ОЗУ, SSD объемом 128 ГБ.
- Клиентские рабочие станции: ПК с ОС Windows 10, 4 ГБ ОЗУ, 2 ГБ свободного места на диске.

5.2 Сетевые требования

- Сеть: Доступ в Интернет со скоростью не менее 10 Мбит/с.
- Сетевые протоколы: Поддержка TCP/IP, HTTPS.

6 Требования к программному обеспечению

6.1 Программные компоненты

- Операционная система: Windows Server.
- Базы данных: MySQL версии не ниже 8.0.
- Программное обеспечение: Веб-сервер Apache 2.4, интерпретатор PHP 8.2 или выше.

6.2 Интерфейсы

- Интерфейс пользователя: Веб-интерфейс с поддержкой браузеров Chrome, Firefox, Edge.

7 Организационно-технические требования

7.1 Этапы разработки

В таблице 1 представлены сроки и этапы разработки веб-приложения.

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						70
Изм.	Лист	№ докум.	Подпись	Дата		

Таблица 1 – Сроки и этапы разработки

Наименование этапов дипломного проекта	Срок
Провести предпроектное исследование.	22.02.2025
Составить техническое задание на разработку программного продукта в соответствии с ГОСТ.	28.02.2025
Провести проектирование программного продукта.	15.03.2025
Разработать программный продукт.	30.04.2025
Выполнить тестирование и отладку программного продукта.	06.05.2025
Разработать документы для программного продукта.	16.05.2025
Составить пояснительную записку.	18.05.2025

Приложение Б код клиентской части

```
<!DOCTYPE html>
<html lang="ru">
<head>
<meta charset="UTF-8">
<title>Админка</title>
<script src="https://cdn.tailwindcss.com"></script>
<meta name="csrf-token" content="{ { csrf_token() } }">
@vite('resources/css/app.css')
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<link rel="shortcut icon" href="{ { asset('images/logo.png') } }" type="image/x-icon">
<script      src="https://api-maps.yandex.ru/2.1/?apikey=fa8f0187-f7ba-4f94-aa91-
c6a61473cec3&lang=ru_RU " type="text/javascript"></script>
</head>
<body class="bg-gray-100 text-gray-900">
<nav class="max-w-7xl mx-auto p-6">
<div class="flex justify-between items-center md:hidden">
<span class="text-xl font-bold">Меню</span>
<button id="menu-toggle" class="text-gray-900 focus:outline-none">
<svg xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24" stroke-
width="1.5" stroke="currentColor" class="w-6 h-6">
<path stroke-linecap="round" stroke-linejoin="round" d="M3.75 6.75h16.5M3.75
12h16.5m-16.5 5.25h16.5" /></svg>
</button>
</div>
<div class="hidden md:flex md:space-x-6 lg:space-x-8 text-sm font-medium">
```

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						72
Изм.	Лист	№ докум.	Подпись	Дата		


```

<a class="p-2 text-black hover:underline" href="{{ route('admin.references.index')
}}">Референсы</a>
<a class="p-2 text-black hover:underline" href="{{ route('admin.car-structure.index')
}}">Структура</a>
<a class="p-2 text-black hover:underline" href="{{ route('admin.equipments.index')
}}">Комплектации</a>
<a class="p-2 text-black hover:underline" href="{{ route('admin.cars.index')
}}">Авто</a>
<a class="p-2 text-black hover:underline" href="{{ route('admin.managers.index')
}}">Управление менеджерами</a>
<a class="p-2 text-black hover:underline" href="{{ route('admin.bookings.index')
}}">Управление бронированиями</a>
<a class="p-2 text-black hover:underline" href="{{ route('dashboard') }}">Личный
кабинет</a>

```

```

</div>

```

```

<div id="mobile-menu" class="hidden md:hidden mt-4 space-y-2 text-sm font-
medium flex flex-col">

```

```

<a class="p-2 text-black hover:bg-gray-200 rounded" href="{{
route('admin.references.index') }}">Референсы</a>
<a class="p-2 text-black hover:bg-gray-200 rounded" href="{{ route('admin.car-
structure.index') }}">Структура</a>
<a class="p-2 text-black hover:bg-gray-200 rounded" href="{{
route('admin.equipments.index') }}">Комплектации</a>
<a class="p-2 text-black hover:bg-gray-200 rounded" href="{{
route('admin.cars.index') }}">Авто</a>
<a class="p-2 text-black hover:bg-gray-200 rounded" href="{{
route('admin.managers.index') }}">Управление менеджерами</a>
<a class="p-2 text-black hover:underline" href="{{ route('admin.bookings.index')
}}">Управление бронированиями</a>

```

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						73
Изм.	Лист	№ докум.	Подпись	Дата		

```

<a class="p-2 text-black hover:bg-gray-200 rounded" href="{{ route('dashboard')
  }}">Личный кабинет</a>
</div>
</nav>
<div class="max-w-7xl mx-auto p-6">
  @yield('content')
</div>
<script>
const menuToggle = document.getElementById('menu-toggle');
const mobileMenu = document.getElementById('mobile-menu');
menuToggle.addEventListener('click', () => {
mobileMenu.classList.toggle('hidden');
});
</script>
</body>
</html>

```

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						74
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение В код серверной части

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\Car;
use App\Models\Brand;
use App\Models\CarModel;
use App\Models\Generation;
use App\Models\Equipment;
use App\Models\BodyType;
use App\Models\EngineType;
use App\Models\TransmissionType;
use App\Models\DriveType;
class CatalogController extends Controller
{
    public function index(Request $request)
    {
        $query = Car::with([
            'equipment.generation.carModel.brand',
            'equipment.bodyType',
            'equipment.engineType',
            'equipment.transmissionType',
            'equipment.driveType'
        ])->where('is_sold', false);
        if ($request->search) {
            $query->where(function($q) use ($request) {
                $q->whereHas('equipment.generation.carModel.brand', function($q) use ($request) {
                    $q->where('name', 'like', "%{$request->search}%");
                })->orWhereHas('equipment.generation.carModel', function($q) use ($request) {
```

					ДП.09.02.07-3.25.211.04. ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

```

$q->where('name', 'like', "%{$request->search}%");
})->orWhereHas('equipment.generation', function($q) use ($request) {
    $q->where('name', 'like', "%{$request->search}%");
});
});
}

if ($request->brand) {
    $query->whereHas('equipment.generation.carModel.brand', function($q) use ($request) {
        $q->where('id', $request->brand);
    });
}

if ($request->model) {
    $query->whereHas('equipment.generation.carModel', function($q) use ($request) {
        $q->where('id', $request->model);
    });
}

if ($request->generation) {
    $query->whereHas('equipment.generation', function($q) use ($request) {
        $q->where('id', $request->generation);
    });
}

if ($request->equipment) {
    $query->where('equipment_id', $request->equipment);
}

if ($request->body_type) {
    $query->whereHas('equipment.bodyType', function($q) use ($request) {
        $q->where('id', $request->body_type);
    });
}

```

```

});
}
if ($request->transmission_type) {
$query->whereHas('equipment.transmissionType', function($q) use ($request) {
$query->where('id', $request->transmission_type);
});
}
if ($request->engine_type) {
$query->whereHas('equipment.engineType', function($q) use ($request) {
$query->where('id', $request->engine_type);
});
}
if ($request->drive_type) {
$query->whereHas('equipment.driveType', function($q) use ($request) {
$query->where('id', $request->drive_type);
});
}
if ($request->min_price) {
$query->where('price', '>=', $request->min_price);
}
if ($request->max_price) {
$query->where('price', '<=', $request->max_price);
}
if ($request->min_mileage) {
$query->where('mileage', '>=', $request->min_mileage);
}
if ($request->max_mileage) {
$query->where('mileage', '<=', $request->max_mileage);
}
}

```

```

if ($request->min_year) {
$query->whereHas('equipment.generation', function($q) use ($request) {
$query->where('year_from', '>=', $request->min_year);
});
}
if ($request->max_year) {
$query->whereHas('equipment.generation', function($q) use ($request) {
$query->where('year_from', '<=', $request->max_year);
});
}
if ($request->min_engine_volume) {
$query->whereHas('equipment', function($q) use ($request) {
$query->where('engine_volume', '>=', $request->min_engine_volume);
});
}
if ($request->max_engine_volume) {
$query->whereHas('equipment', function($q) use ($request) {
$query->where('engine_volume', '<=', $request->max_engine_volume);
});
}
$sort = $request->sort ?? 'price';
$direction = $request->direction ?? 'asc';
$query->orderBy(
match($sort) {
'year' => Equipment::select('generations.year_from')
->join('generations', 'generations.id', '=', 'equipment.generation_id')
->whereColumn('equipment.id', 'cars.equipment_id'),
'mileage' => 'mileage',
'price' => 'price',

```

```

'engine_volume' => Equipment::select('engine_volume')
->whereColumn('equipments.id', 'cars.equipment_id'),
default => $sort
},
$direction
);
$cars = $query->paginate(12);
$brands = Brand::has('models.generations.equipments.cars')->get();
$models = CarModel::when($request->brand, function($q) use ($request) {
$q->where('brand_id', $request->brand);
})->has('generations.equipments.cars')->get();
$generations = Generation::when($request->model, function($q) use ($request) {
$q->where('car_model_id', $request->model);
})->has('equipments.cars')->get();
$equipments = Equipment::when($request->generation, function($q) use ($request) {
$q->where('generation_id', $request->generation);
})->has('cars')->get();
$bodyTypes = BodyType::has('equipments.cars')->get();
$engineTypes = EngineType::has('equipments.cars')->get();
$transmissionTypes = TransmissionType::has('equipments.cars')->get();
$driveTypes = DriveType::has('equipments.cars')->get();
return view('catalog.index', compact(
'cars',
'brands',
'models',
'generations',
'equipments',
'bodyTypes',
'engineTypes',

```

```

'transmissionTypes',
'driveTypes'
));
}

public function show(Car $car)
{
$car->load([
'equipment.generation.carModel.brand',
'equipment.bodyType',
'equipment.driveType',
'equipment.engineType',
'equipment.transmissionType',
'equipment.country',
'branch',
'color',
'images'
]);
$has3dModel = $car->equipment->model_url !== null;
return view('cars.show', compact('car', 'has3dModel'));
}
}

```