|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н. Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_\_***ИУК «Информатика и управление»\_\_\_\_\_\_\_\_\_***

**КАФЕДРА** \_\_ ***ИУК5 «Системы обработки информации» \_\_***

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

***Разработка веб-приложения для сети автомоек***

по дисциплине ***Базы данных***

Студент гр.\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Макаренко С.С.)

(подпись) (Ф.И.О.)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Кириллов В.Ю.)

(подпись) (Ф.И.О.)

Оценка руководителя \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30–50 (дата)

Оценка защиты \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_

30–50 (дата)

Оценка проекта \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка по пятибалльной шкале)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Калуга, 2021

Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования

***«Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н. Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой **\_\_ИУК5\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_ (Е.В. Вершинин)

«10» сентября 2021г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине **Базы данных**

Студент Макаренко С.С. ИУК5-51Б\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ \_\_\_\_

(фамилия, инициалы, индекс группы)

Руководитель Кириллов В.Ю.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы)

График выполнения проекта: 25% к 4 нед., 50% к 7 нед., 75% к 10 нед., 100% к 14 нед.

***1. Тема курсового проекта***

***Разработка веб-приложения для сети автомоек***

***2. Техническое задание***

Разработать веб-приложения для сети автомоек

***3. Оформление курсового проекта***

3.1. Расчетно-пояснительная записка на \_\_\_ листах формата А4.

3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.)\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

Дата выдачи задания «10» сентября 2021г.

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_/\_\_\_\_ Кириллов В.Ю. \_

(подпись) (Ф.И.О.)

Задание получил\_\_\_\_\_\_\_\_\_\_\_\_\_ \_/\_Макаренко С.С.\_\_/ «10» сентября\_2021 г. \_

(подпись) (Ф.И.О.)

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 3](#_Toc89966309)

[1. Техническое задание 4](#_Toc89966310)

[2. Научно-исследовательская часть 11](#_Toc89966311)

[2.1. Постановка задачи проектирования. 11](#_Toc89966312)

[2.2. Описание предметной области. 11](#_Toc89966313)

[2.3. Анализ аналогов и прототипов. 12](#_Toc89966314)

[2.4. Перечень задач, подлежащих решению в процессе разработки. 18](#_Toc89966315)

[2.5. Обоснование выбора инструментов и платформы для разработки. 19](#_Toc89966316)

[3. Проектно-конструкторская часть 24](#_Toc89966317)

[3.1. Разработка структуры системы. 24](#_Toc89966318)

[3.2. Разработка алгоритмов обработки информации. 27](#_Toc89966319)

[3.3. Разработка архитектуры приложения. 28](#_Toc89966320)

[3.4. Реализация готового приложения. 29](#_Toc89966321)

[3.5. Разработка интерфейса взаимодействия пользователя с системой. 33](#_Toc89966322)

[4. Проектно-технологическая часть 42](#_Toc89966323)

[4.1. Тестирование и отладка макета рабочей программы 42](#_Toc89966324)

[4.2. Разработка руководства программиста и работника автомойки. 48](#_Toc89966325)

[5. Вывод 52](#_Toc89966326)

[6. Список использованных источников 53](#_Toc89966327)

1. Техническое задание

1. Общие сведения

1.1 Полное наименование системы

Веб-приложение для сети автомоек, позволяющее выбирать филиал, услуги и время записи.

1.2 Наименования предприятий разработчика и заказчика

Разработчик: студент группы ИУК5-51Б Макаренко С.С.

Заказчик: КФ МГТУ им.Баумана кафедра ИУК5

1.3 Основания для разработки

Основанием для разработки веб-приложения для сети автомоек является задание в соответствии с учебным планом дисциплины «Базы данных»

1.4 Перечень документов, на основании которых создается система

Методические указания к выполнению курсовой работы по предметам “Базы данных” и “Архитектура автоматизированных систем обработки информации и управления”

1.5 Плановые сроки начала и окончания работы по созданию системы

Срок начала работ: 06.09.2021

Срок окончания работ: 10.12.2021

1.6. Сведения об источниках и порядке финансирования работ

Источников и порядков финансирования не имеется

2. Назначение и цели создания (развития) системы

2.1 Назначение системы.

Разрабатываемая система предназначена для крупной сети автомоек. Её основные назначения – избавиться от очередей, разгрузить менеджеров, сделать равномерную, полную загрузку автомоек и обеспечить удобство использования для клиентов.

2.2 Цели создания системы.

Целями создания системы являются:

* реализация основных назначений системы;
* изучение новых возможностей работы с базами данных в сочетании с веб-программированием;
* овладение навыками ведения научно-исследовательской, проектной и производственно-технологической деятельности, развитие творческих способностей;
* усвоение методов грамотного ведения, оформления и редактирования технической документации.

3.Характеристики объекта автоматизации

3.1 Краткие сведения об объекте автоматизации.

Объектом автоматизации является процесс записи на автомойку, т.е. автоматическое предложение списка филиалов, доступных услуг в выбранном филиале и свободного времени.

4. Требования к системе

4.1 Требования к системе в целом

Программный продукт должен представлять собой веб-приложение с продуманной логикой, сервер которого находится на локальном компьютере.

4.1.1 Требования к структуре и функционированию системы

Приложение должно быть разбито на 3 части:

- Графический пользовательский интерфейс. Визуализация данных в понятном для пользователя виде.

- Сервер. Обработка запросов от графического интерфейса, обращение к БД в соответствии с содержанием запроса и возврат результатов из БД.

- База данных. Запись, хранение, удаление, изменение и представление информации в требуемом виде.

4.1.2 Требования к надежности

Программа должна обеспечивать корректную обработку исключительных ситуаций и оповещать пользователя о них.

4.2 Требования к функциям, выполняемым системой

4.2.1 Функционал для пользователя:

1. Регистрация и авторизация;
2. Выбор филиала автомойки, услуг и времени записи;
3. Автоматический расчёт суммарного времени выбранных услуг;
4. Автоматический расчёт итоговой стоимости выбранных услуг, с применением скидки пользователя (скидка пользователя рассчитывается в соответствии с общей суммой услуг, ранее оказанных пользователю);
5. Просмотр своих записей, с возможностью отмены.

4.2.2 Функционал для администратора:

1. Просмотр всех записей, с использованием фильтров для поиска;
2. Добавление филиалов автомоек;
3. Добавление услуг, с указанием времени и стоимости;
4. Добавление услуги в филиал;

4.2.3 Функционал для работника:

1. Просмотр всех записей на текущий день, на выбранную автомойку;
2. Изменение статуса услуг (создана, выполнена, отменена, пропущена);
3. Регистрация и запись нового клиента.

4.3 Требования к видам обеспечения

4.3.1 Требования к прикладным программам

Для работы программного комплекса необходимы:

* версия браузера не ниже:
* Chrome: 64
* Firefox: 58
* Safari: 11
* Opera: 51
* Edge: 42

4.3.2 Требования к составу технических средств

Для функционирования системы необходимы:

* устройство с подключением к интернету.

5. Состав и содержание работ по созданию системы

Таблица 1. Требования и сроки исполнения.

|  |  |
| --- | --- |
| Требования | Сроки исполнения |
| **1** | **2** |
| Утверждение темы, задания на разработку, технического задания; | 10.09.2021 |

Продолжение таблицы 1.

|  |  |
| --- | --- |
| **1** | **2** |
| Оформление ТЗ. Описание предметной области и требований к системе. Аналоги. Концептуальная схема. Прототип и скетчи интерфейса; | 01.10.2021 |
| Оформление введения и исследовательской части. Обоснование выбора БД (если необходимо), логическая схема БД, физическая схема данных. Демонстрация работы макета системы с БД; | 23.10.2021 |
| Оформление проектно-конструкторской части.  Демонстрация работающего приложения. Презентация. Тестирование и отладка приложения. Разработка эксплуатационной документации. | 15.11.2021 |
| Все ошибки и проблемы устранены. Демонстрация проекта. Защита. | 10.12.2021 |

6. Порядок контроля и приемки системы.

6.1 Состав, объем и методы испытаний системы и ее составных частей.

Приложение должно пройти предварительные испытания, состоящие из отладки и дымового тестирования.

По итогам предварительных испытаний должны быть внесены исправления, учитывающие замечания, полученные в ходе предварительных испытаний.

Для проверки результата внесенных изменений должны быть проведены повторные испытания по ранее разработанной программе.

6.2. Общие требования к приемке работ.

В процессе приёмки работы должна быть осуществлена проверка на соответствие требованиями настоящего «Технического задания». По результатам испытаний возможны доработки и исправления.

При работе осуществляется проверка на соответствие требований актуального технического задания «Технического задания». По результатам можно исправить недочёты и внести некоторые доработки.

Для проверки корректной работы внесённых изменений должны быть проведены повторные испытания разработанной программы.

7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Для подготовки объекта автоматизации к вводу системы в действие должны быть проведены следующие мероприятия:

1. Обеспечить пригодность для обработки, информации поступающей в систему;
2. Проанализировать нововведения, которые необходимо осуществить в объекте автоматизации;
3. Задать, необходимые для функционирования условия, при которых, разрабатываемая система будет функционировать согласно требованиям ТЗ.

8. Требования к документированию

По окончанию работы предъявлена расчетно-пояснительная записка в состав которой входят:

* техническое задание;
* научно-исследовательская часть;
* проектно-конструкторская часть;
* проектно-технологическая часть.

Также должна быть предоставлена графическая часть работы, выполненная формате А1 на 2 листах, в которую входят:

* демонстрационные чертежи;
* алгоритмические схемы.

9. Источники разработки

1.Гост 34.601-90

2.Гост 34.602-89

1. Научно-исследовательская часть
2. Постановка задачи проектирования.

Задачей проектирования данной курсовой работы является разработка веб-приложения для сети автомоек, предназначенного для записи. Так же необходимо разработать удобный пользовательский интерфейс для работы с приложением.

1. Описание предметной области.

Автомойка – организация, осуществляющая мойку автомобилей и оказывающая сопутствующие услуги.

В городе Х есть сеть автомоек (5-7 шт). Проблема неравномерности загрузки присутствует, как в каждом филиале, так и в сети автомоек в целом. Например, в филиале №1 стоит огромная очередь автомобилей, а в филиале №2, который находится неподалёку машин нет вообще. И также в каждом филиале бывают «часы пик» в которые скапливается большая очередь, а потом нет клиентов вообще. Для решения данной проблемы руководство принимает решение о создании сайта, в котором пользователи будут записываться на автомойку, и приезжать к назначенному времени.

Необходим именно сайт т.к. его разработка дешевле и быстрее, чем написание кроссплатформенных приложений. Также любое приложение необходимо установить и скачать, а сайт – нет. Также сайт будет заодно и рекламировать сеть автомоек.

Отслеживать очередь в режиме реального времени нет необходимости, т.к. при нажатии кнопки записаться будет видно ближайшее время.

У каждой записи в автомойку есть: только один пользователь, совершивший эту запись; только один филиал автомойки, выбранный пользователем; дата и время; одна или множество услуг, выбранных пользователем; стоимость услуг; длительность услуг; статус (создана – стандартный статус после создания записи, выполнена – услуги оказаны и оплачены, отменена – клиент отменил запись, пропущена – если клиент не явился в назначенное время). У каждого филиала автомойки есть свой адрес. У каждой услуги есть название, её цена и время оказания. У филиала может временно не быть услуг, также и у услуги может временно не быть филиалов, где её оказывают. У каждой услуги есть длительность и цена, соответственно, при выборе нескольких услуг суммируется их стоимость и время выполнения. Каждый пользователь имеет скидку, на основании суммы его заказов.

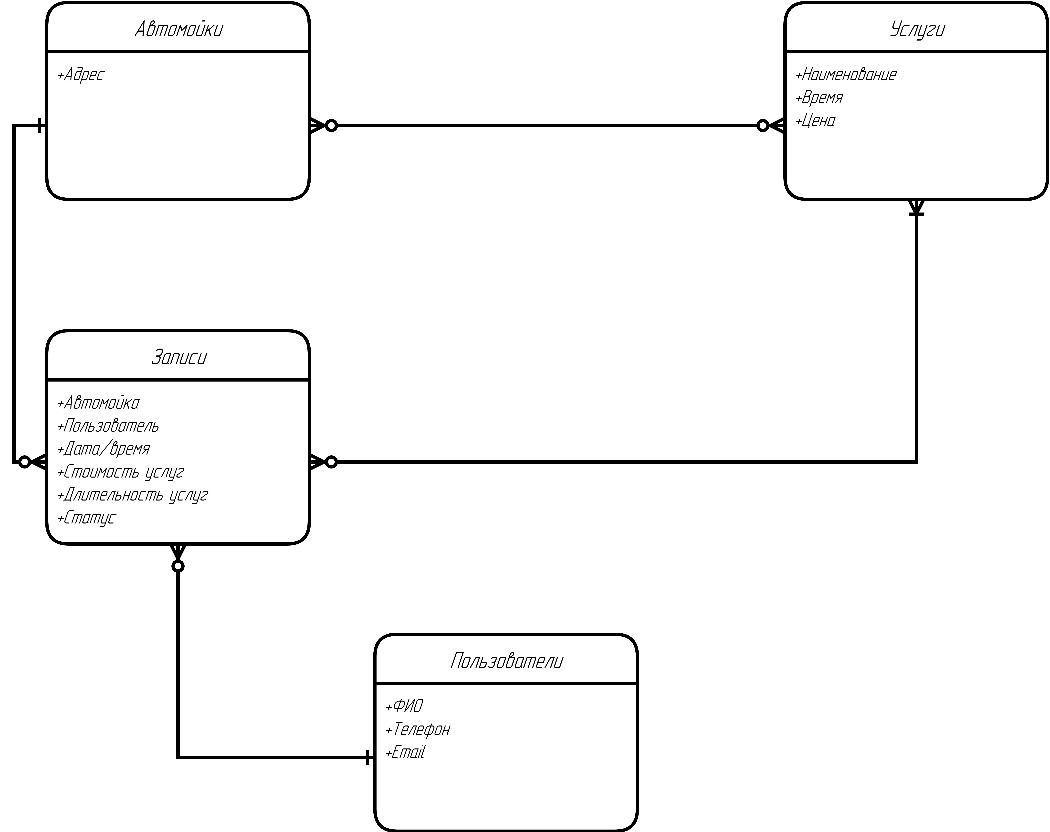
**

Рисунок 1. Концептуальная схема.

1. Анализ аналогов и прототипов.

У некоторых автомоек уже существуют сайты, на которых пользователю предлагают записаться на обслуживание, разберём их плюсы и минусы:

1. «formula76»

Ссылка на сайт: <https://formula76.ru/tireservice/washrecord>

Сеть автосервисов «Формула» — многофункциональный центр по обслуживанию автомобилей в Ярославле. У них есть 3 филиала, где можно вымыть автомобиль, выполнить ремонтные работы, купить шины, диски и АКБ.

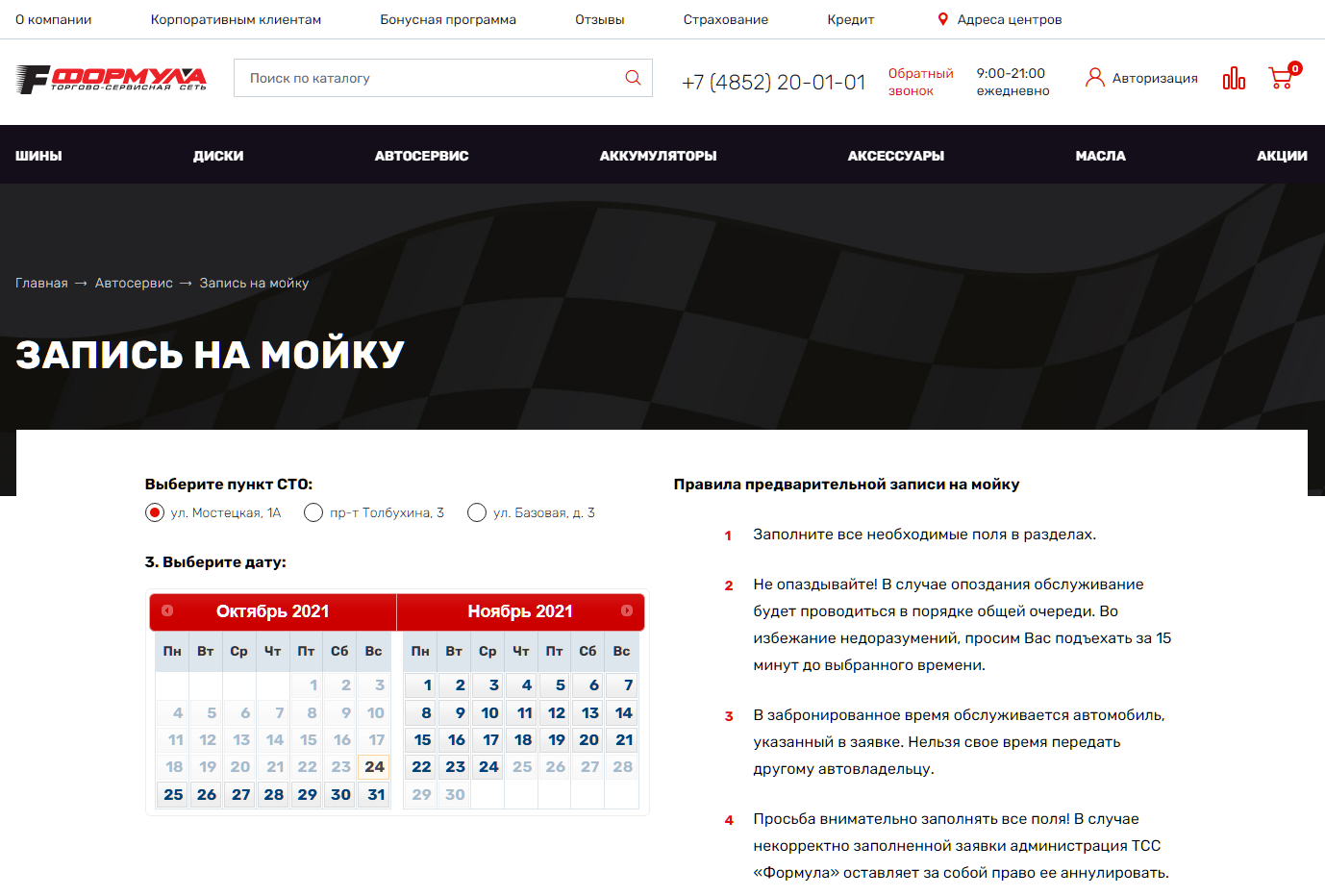
**

Рисунок 2. Веб-сайт «formula76».

Плюсы:

* Возможность записаться на автомойку без регистрации и авторизации на сайте;
* Удобный выбор даты.

Минусы:

* Часовые промежутки во времени, что крайне неудобно в текущих реалиях мира, когда у некоторых людей день расписан буквально по минутам.
* Нет списка оказываемых услуг, и тем более их стоимости и времени выполнения. Соответственно человек, который зашёл на этот сайт не имеет возможности узнать, выполняется ли интересующая его услуга на данной автомойке или нет. Также любой человек, прежде чем записаться, хочет узнать цену, которую ему придётся заплатить.

1. Автотехцентр «Гараж»

Ссылка на сайт: <http://atc-garage.ru>

«Гараж» – это сеть многофункциональных технических центров премиум класса, оказывающих полный спектр услуг по первоклассному обслуживанию автомобилей. Этот автотехцентр является лидером на рынке авторемонтных, автомоечных и автокосметических услуг Калужской области.

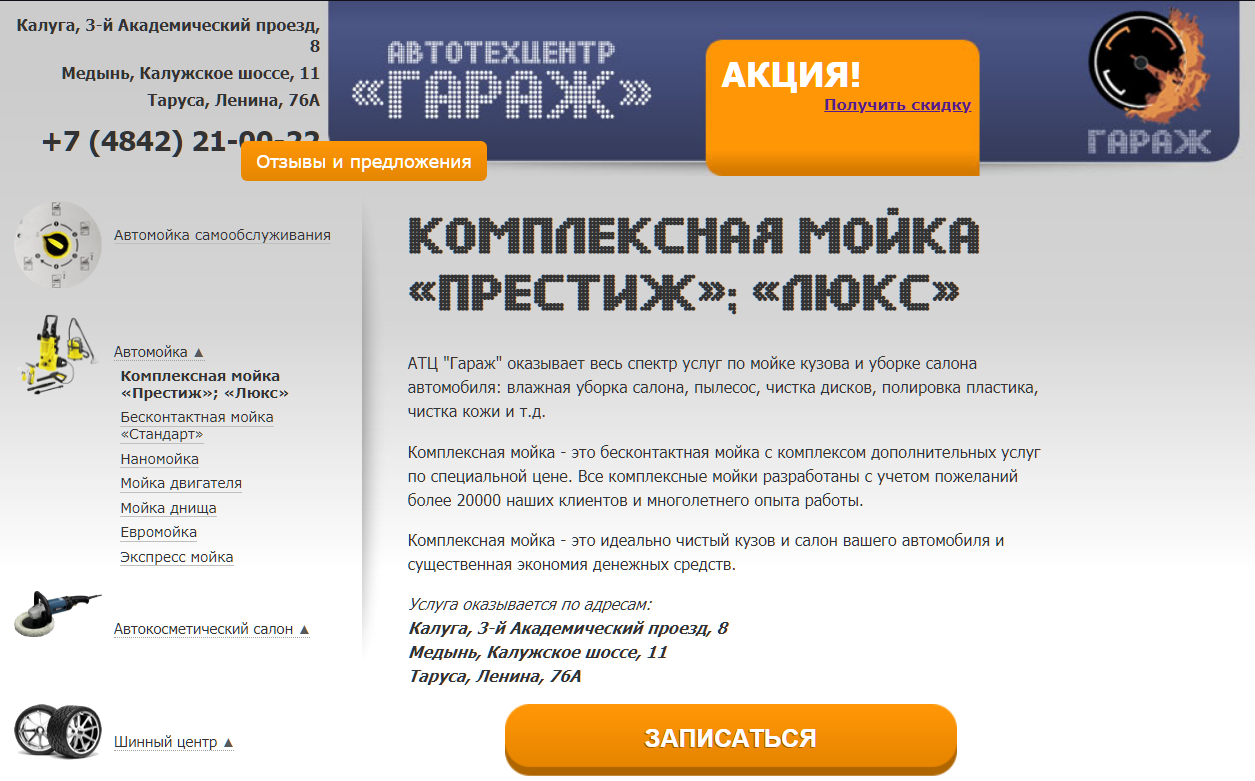


Рисунок 3. Веб-сайт «atc-garage».

Плюсы:

* Возможность записаться на автомойку без регистрации и авторизации на сайте

Минусы:

* Получасовые интервалы, это уже лучше, чем в прошлом примере, но всё же недостаточно удобно
* При записи можно выбрать только один тип мойки
* Как и в прошлом примере неизвестны цена и длительность оказания услуги

1. «RocketWash»

Ссылка на сайт: <https://www.rocketwash.me/avtolyubitelyam/mobile-app>

Фактически, это не автомойка, а CRM-система для автомоек, которая призвана автоматизировать все процессы. Но т.к. этот сайт выдаётся поисковиком по запросу «Записаться на автомойку», мы всё равно его рассмотрим. Конкретно на сайте, нет возможности записаться на автомойку, есть только ссылки на приложения в AppStore и GooglePlay. Это не совсем подходит для темы курсовой работы, но всё же для исследования предметной области, скачаем приложение.

При запуске приложения мы видим автомойки подключенные, к данной CRM-системе (Рисунок 4). При выборе автомойки мы видим ближайшее свободное время и примерные цены на услуги (Рисунок 5). Нажмём кнопку «Записаться», приложение требует указать марку и модель авто, именно от этого будут зависеть точные цены на услуги (полезная фишка, но в нашем случае – не обязательная). Далее мы видим список услуг с указанием их цены, и при нажатии на значок вопроса появляется дополнительная информация по услуге

|  |  |
| --- | --- |
| map  Рисунок 4. Карта в мобильном приложении «RocketWash». | 2 |
|  | Рисунок 5. Выбрана автомойка. |

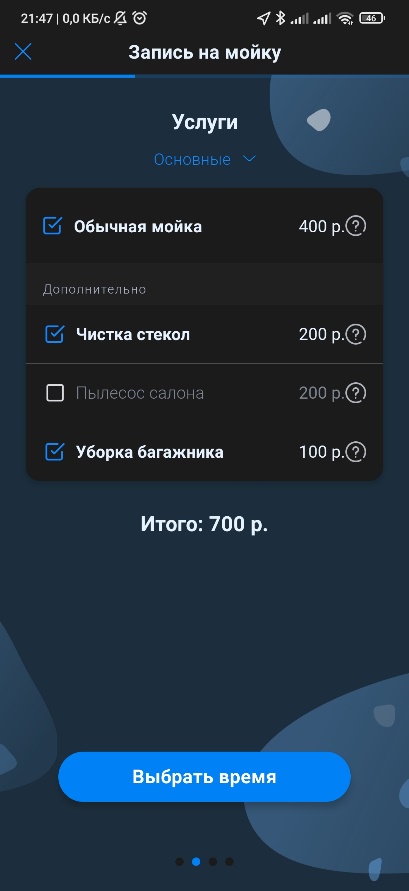
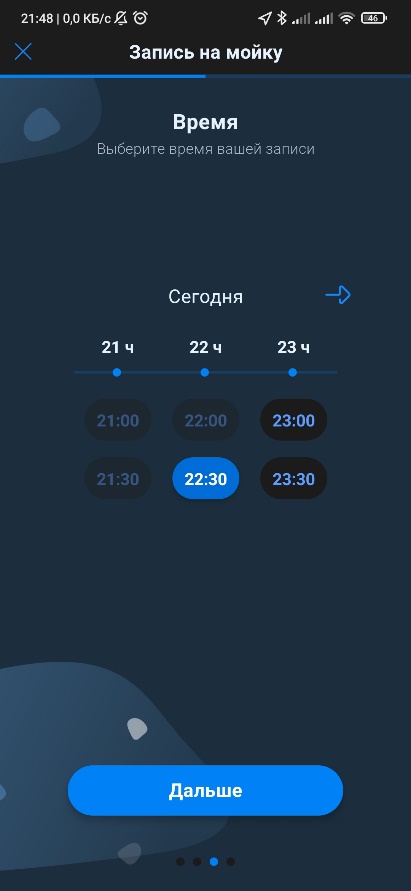
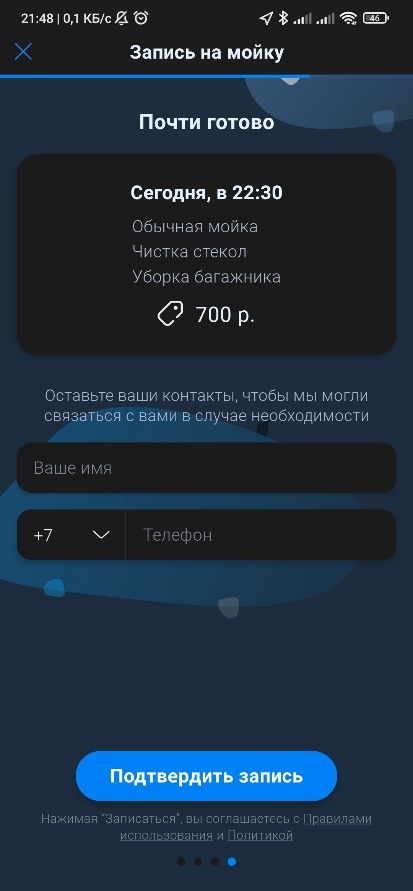
  

Рисунок 6. Остальные этапы записи на автомойку в приложении «RocketWash».

Плюсы:

* Информация о стоимости каждой услуги
* Выбор списка необходимых услуг с подсчётом итоговой цены
* Возможность записаться на автомойку без регистрации и авторизации в приложении

Минусы:

* Необходимость скачивания приложения
* Получасовые интервалы
* Нет информации по длительности оказания услуг

Естественно, здесь были рассмотрены далеко не все аналоги, но минусы везде примерно одинаковые.

Я в свое приложение возьму такие достоинства аналогов:

* информация о стоимости каждой услуги;
* выбор списка необходимых услуг с подсчётом итоговой цены;
* возможность записаться на автомойку без авторизации с помощью быстрой регистрации при записи;
* удобный выбор даты, в виде календаря.

и избегну таких недостатков:

* необходимость скачивания приложения;
* неудобные временные интервалы;
* отсутствие информации по длительности оказания услуг;
* при записи можно выбрать только один тип мойки.

1. Перечень задач, подлежащих решению в процессе разработки.

Исходя из результатов анализа существующих аналогов было принято решение разработать собственное веб-приложение для сети автомоек.

Интерфейс пользователя должен предоставлять пользователю функционал:

* выбор филиала автомойки;
* выбор необходимых услуг из списка;
* просмотр цены и длительности каждой услуги;
* просмотр итоговой стоимости и времени оказания выбранных услуг;
* удобный выбор даты для записи;
* запись с временными интервалами в 5 минут (можно сделать и поминутную запись, но в этом список свободного времени будет слишком большой, что будет неудобно для пользователей);
* просмотр своих записей;
* отмена записи;
* смена своего пароля.

Интерфейс работника должен предоставлять функционал:

* регистрация и запись нового клиента;
* выбор автомойки и просмотр записей на выбранную автомойку на текущий день;
* изменение статуса записи;
* смена своего пароля.

Интерфейс администратора должен предоставлять функционал:

* добавление и удаление филиалов автомойки;
* добавление услуг с указанием цены и длительности оказания;
* добавление услуг в филиал;
* просмотр всех записей с использованием фильтрации;
* изменение статуса записи;
* смена своего пароля.

В процессе разработки должен быть выделен следующий перечень задач:

* Разработать удобный и интуитивно-понятный интерфейс веб-приложения с вышеперечисленным функционалом;
* Реализовать всплывающие окна с предупреждениями при неверных действиях пользователя;
* При реализации необходимо предусмотреть проверку на корректность введенных данных;

Также нужно протестировать и отладить все этапы процесса разработки ПО.

1. Обоснование выбора инструментов и платформы для разработки.

В качестве среды разработки используется Visual Studio Code. Интегрированная среда разработки Visual Studio Code — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения. Помимо стандартного редактора и отладчика, которые существуют в большинстве сред IDE, Visual Studio Code включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для упрощения процесса разработки.

Согласно опросу, проведенного среди 3240 разработчиков в феврале 2018 организацией Standard, разработчики выделили такие средства разработки, как наиболее часто используемые:

Первое место среди IDE у Visual Studio Code с 16.8% использующих его разработчиков.

Также, Visual Studio Code обладает хорошим отладчиком, с помощью которого можно легко отследить ошибки в коде и без проблем их исправить, позволяет удобно и быстро создавать проекты и работать с ними.

По сравнению со средой разработки Eclipse, Visual Studio Code имеет более простой интерфейс в использовании.

В качестве платформы разработки будет использоваться Node.js.

Node.js – программная платформа, основанная на движке V8, превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения (при помощи NW.js, AppJS или Electron для Linux, Windows и Mac OS) и даже программировать микроконтроллеры (например, tessel и espruino). В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом. Популярность в области построения web-сайтов определяется наличием большого набора встроенных средств и крупнейшего менеджера подключаемых пакетов npmjs.com для разработки web-приложений.

Основные из встроенных средств и подключаемых пакетов:

* извлечение POST и GET-параметров, а также переменных окружения web-сервера в предопределённые массивы;
* отправка HTTP-заголовков;
* работа с cookies и сессиями;
* обработка файлов, загружаемых на сервер;
* работа с HTML заголовками и HTTP авторизацией.

3) Как можно догадаться из используемой платформы, в качестве языка программирования будет использоваться JavaScript.

JavaScript – объектно-ориентированный скриптовый язык программирования. Является диалектом языка ECMAScript.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.

Для создания интерактивного пользовательского интерфейса будут использоваться библиотеки React и React-Bootstrap.

React – JavaScript-библиотека с открытым исходным кодом для разработки пользовательских интерфейсов. Его главная задача – обеспечение вывода на экран того, что можно видеть на веб-страницах. React значительно облегчает создание интерфейсов благодаря разбиению каждой страницы на небольшие фрагменты.

React-Bootstrap – это библиотека React-компонентов, подходящих для повторного использования, которая реализует возможности популярного шаблона Bootstrap. А Bootstrap в свою очередь, это – свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя HTML- и CSS-шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения.

В качестве базы данных (далее по тексту – БД) была выбрана PostgreSQL.

PostgreSQL не просто реляционная, а объектно-реляционная СУБД. Это даёт ему некоторые преимущества над другими SQL базами данных с открытым исходным кодом, такими как MySQL, MariaDB и Firebird.

Фундаментальная характеристика объектно-реляционной базы данных – это поддержка пользовательских объектов и их поведения, включая типы данных, функции, операции, домены и индексы. Это делает Постгрес невероятно гибким и надежным. Среди прочего, он умеет создавать, хранить и извлекать сложные структуры данных.

Для работы с базой данных будет использоваться Sequalize.

Sequelize - это ORM-библиотека для приложений на Node.js, которая осуществляет сопоставление таблиц в бд и отношений между ними с классами. При использовании Sequelize мы можем не писать SQL-запросы, а работать с данными как с обычными объектами. Причем Sequelize может работать с рядом СУБД - MySQL, Postgres, MariaDB, SQLite, MS SQL Server.

В итоге, разработка приложения будет производиться в среде Visual Studio Code, на платформе Node.js, на языке JavaScript с использованием библиотек React, React-Bootstrap и Sequelize. В качестве БД будет использоваться PostgreSQL. Также естественно будут использоваться HTML и CSS.

1. Проектно-конструкторская часть
2. Разработка структуры системы.

Исходя из описания предметной области и функционала, необходимого заказчику, составим логическую схему базы данных.

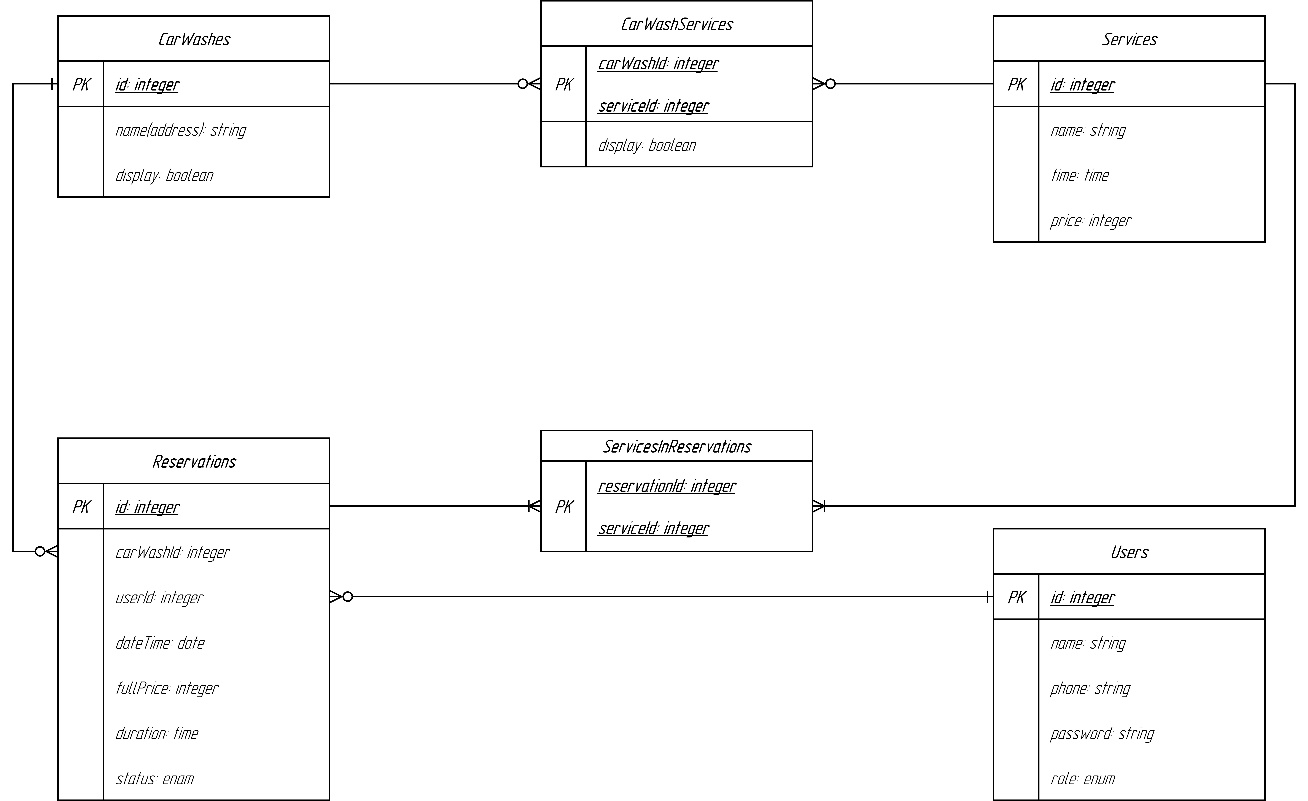
**

Рисунок 7. Логическая схема базы данных.

Рассмотрим каждую таблицу(сущность):

1. CarWashes

Назначение: хранение и представление списка автомоек.

Атрибуты: id – уникальный номер для однозначной идентификации автомойки; name(address) – название автомойки, т.к. все автомойки – филиалы одной сети, уникальных названий у них нет, поэтому вводится просто адрес филиала; display – отвечает за отображение филиала в пользовательском интерфейсе (по умолчанию равен “true”), при “удалении” филиала автомойки значение становится false и филиал автомойки перестаёт отображаться в пользовательском интерфейсе.

1. Services

Назначение: хранение и представление списка всех услуг.

Атрибуты: id – уникальный номер для однозначной идентификации услуги; name – наименование услуги; time – длительность оказания услуги; price – стоимость услуги в рублях.

1. CarWashServices

Назначение: реализация связи многие-ко-многим, то есть в одном филиале может оказываться несколько услуг, и каждая услуга может оказываться в разных филиалах; хранение и представление соответствия услуг и филиалов автомойки.

Атрибуты: carWashId и serviceId – уникальный составной ключ для обеспечения семантической целостности, где carWashId – это id автомойки, а serviceId – id услуги; display – отвечает за отображение соответствия филиала и услуги в пользовательском интерфейсе (по умолчанию равен “true”), при “удалении” услуги из филиала значение становится false и данная услуга перестаёт отображаться в данном филиале в пользовательском интерфейсе.

1. Users

Назначение: хранение и представление списка пользователей.

Атрибуты: id – уникальный номер для однозначной идентификации пользователя; name – может быть, как полным ФИО, так и просто именем; phone – номер телефона пользователя, используется для авторизации, и как контактные данные пользователя; password – пароль, указанный пользователем при регистрации, используется для авторизации; role – право доступа для пользователя, имеет тип enum со значениями 'ADMIN' (администратор сети автомоек), 'WORKER' (работник), 'USER' (клиент).

1. Reservations

Назначение: хранение и представление списка автомоек.

Атрибуты: id – уникальный номер для однозначной идентификации записи; carWashId – id автомойки; userId – id пользователя, который записан в автомойку; dateTime – дата и время записи в автомойку; fullPrice – итоговая стоимость всех выбранных пользователем услуг, с учётом индивидуальной скидки; duration – суммарная длительность оказания всех выбранных услуг; status – статус записи имеет тип enum со значениями 'создана' (статус по умолчанию после создания записи), 'отменена' (пользователь, администратор или работник автомойки отменил запись), 'пропущена' (пользователь не явился в назначенное время), 'выполнена' (все выбранные услуги оказаны).

1. ServicesInReservations

Назначение: реализация связи многие-ко-многим, то есть в одной записи может быть выбрано несколько услуг, и каждая услуга может быть выбрана в разных записях; хранение и представление соответствия услуг и филиалов автомойки.

Атрибуты: reservationId и serviceId – уникальный составной ключ для обеспечения семантической целостности, где reservationId – это id записи, а serviceId – id услуги; display – отвечает за отображение соответствия филиала и услуги в пользовательском интерфейсе (по умолчанию равен “true”), при “удалении” услуги из филиала значение становится false и данная услуга перестаёт отображаться в данном филиале в пользовательском интерфейсе.

1. Также в базе данных присутствует ещё одна таблица, она не обозначена на схеме, так как она ни к чему не привязана и данные из неё можно с таким же успехом хранить в конфигурационном файле. Рассмотрим и её.

DefineSales

Назначение: хранение и представление информации о скидках.

Атрибуты: id – уникальный номер для однозначной идентификации скидки; dsale – скидка в %; dsum – общая стоимость записей пользователя со статусом 'выполнена', при которой начинает действовать данная скидка.

1. Разработка алгоритмов обработки информации.

В общем виде обработка информации происходит так:

1. пользователь в интерфейсе веб-приложения вводит какие-либо данные, нажимает кнопку;
2. отправляется http запрос с вложенными параметрами на сервер;
3. сервер обрабатывает запрос и отправляет sql запрос в базу данных;
4. база данных возвращает ответ на sql запрос;
5. сервер возвращает строки из базы данных в виде http ответа на клиент;
6. веб-приложение обрабатывает ответ и отображает его в необходимом виде в интерфейсе.

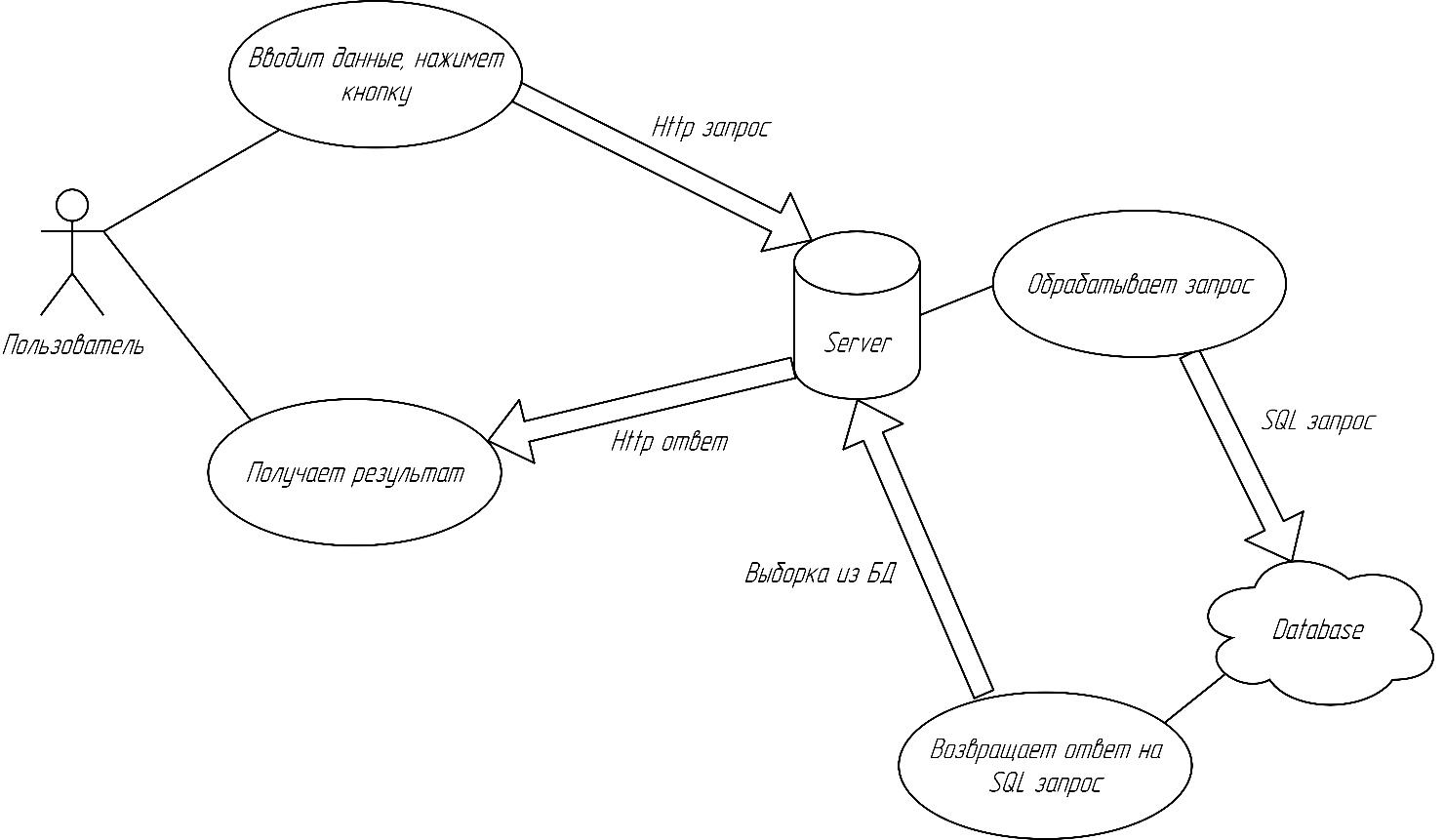
**

Рисунок 8. Алгоритм запросов.

1. Разработка архитектуры приложения.

Для каждого обращения к базе данных на клиентской части приложения необходимо объявить http запрос как константу. Все эти константы запросов для приложения пропишем в файле «applicationAPI.js», а для пользователей в файле «userApi.js» и будем экспортировать эти константы запросов при необходимости.

На сервере создадим «mainRouter.js», который будет принимать http запросы и проверяя путь направлять их в соответствующий роутер.

Каждый роутер в зависимости от типа запроса (get/post/put/delete) и возможного добавочного пути будет вызывать соответствующий метод класса из контроллера.

Также для дополнительной безопасности будут использоваться функции промежуточной обработки (middleware). А именно проверка авторизации пользователя и проверка роли. Это очень важно при добавлении или обновлении информации в базе данных. В middleware ещё будет размещён обработчик ошибок, связанных с запросами к базе данных.

Для аутентификации пользователей будут использоваться jwt токены из пакета «jsonwebtoken».

Пароли в базе данных будут храниться в зашифрованном виде с помощью пакета «bcrypt».

Для удобной работы с базой данных, лёгкому переходу на другую базу данных, возможностью работать с данными, как с обычными объектами будет использоваться ORM-библиотека «sequelize».

На клиентской части для хранения результатов запросов из базы данных будут созданы 2 файла: ApplicationStore.js – для хранения информации о приложении (список филиалов автомойки, список услуг и тд.); и UserStore.js – для хранения информации о пользователе (роль, статус).

Интерфейс веб-приложения должен быть модульным, то есть страницы должны состоять из множества частей, собранных в одно целое. Это обеспечит лёгкое исправление ошибок и простоту модернизации веб-приложения в дальнейшем.

1. Реализация готового приложения.

На сервере создадим файл «db.js», в котором объявим подключение к базе данных с помощью «sequelize». Данные для этого подключения берутся из фала «.env», также в этом файле хранится секретный ключ для создания jwt токена.

Для создания модели базы данных, на сервере была создана папка «models» и в этой папке файл «models.js». В котором с помощью «require» был загружен созданный ранее модуль подключения к базе данных из файла «db.js».

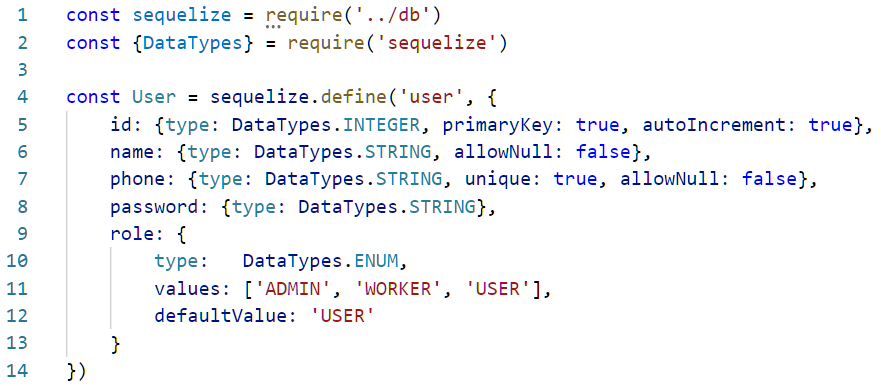


Рисунок 9. Пример создания сущности в базе данных с помощью ORM-библиотеки «sequelize».

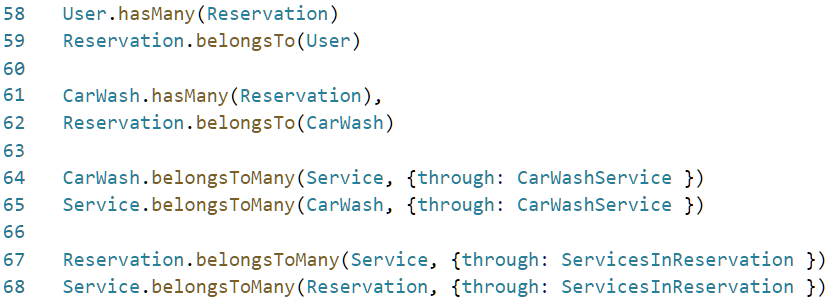


Рисунок 10. Пример описания отношений сущностей в базе данных с помощью ORM-библиотеки «sequelize».

Для удобства и логического разделения кода, каждой сущности будет соответствовать свой роутер и контроллер. Все контроллеры объединены в папку controllers, а все роутеры – соответственно в routes.

В каждом контроллере загружаем соответствующий ему модуль сущности из ранее созданного файла «models.js». И прописываем методы для запросов к базе данных (Рисунок 11). Естественно не все запросы удобно писать через встроенные функции sequelize, а некоторые – просто невозможно. Поэтому в sequelize есть возможность написать запрос вручную (Рисунок 12).

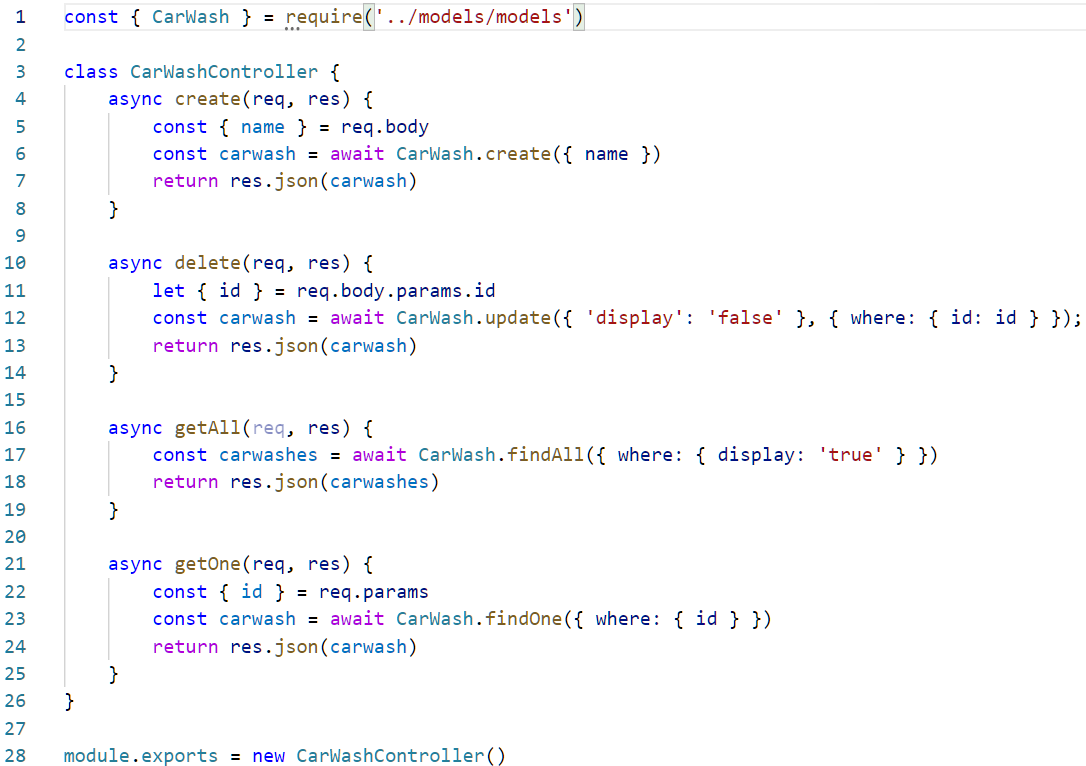


Рисунок 11. Создание контроллера для филиалов автомойки в файле «carWashController.js».

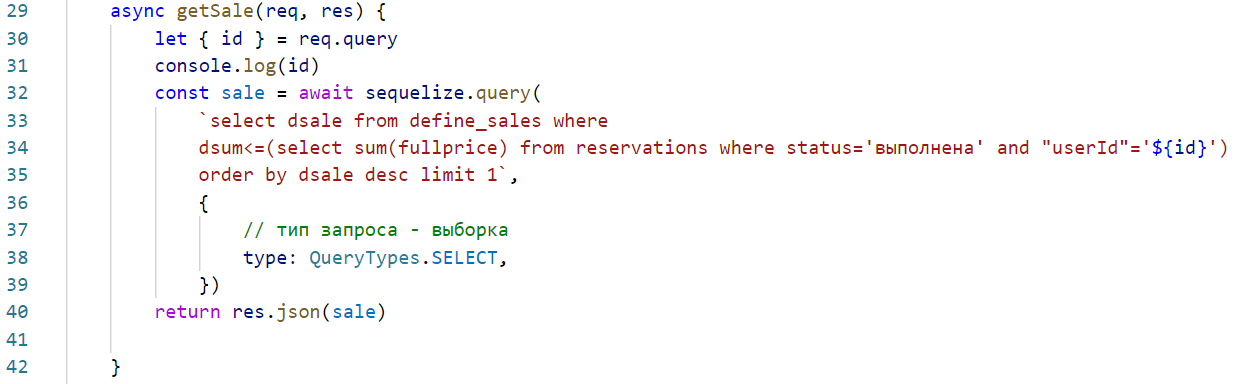


Рисунок 12. Пример написания запроса вручную.

В каждом роутере обрабатываем тип http запроса и добавочный путь, если есть, а также при необходимости используем middleware проверку прав пользователя, отправившего этот запрос. И выбираем подходящий метод из контроллера.



Рисунок 13. Пример обработки http запросов в роутере, файл «carWashRouter.js».



Рисунок 14. Пример первичной обработки http запроса на сервере, и перенаправление в соответствующий ему роутер, файл «mainRouter.js».

На текущем этапе разработки в базе данных не предусмотрено никаких дополнительных проверок и валидаций полей, так как разработка приложения производится одним человеком, который сам и проектировал эту базу данных. Также все необходимые валидации полей, например, формат номера телефона пользователя, незаполненные поля и тд. проверяются с помощью JavaScript. В будущем, при необходимости расширения приложения можно будет добавить дополнительные триггеры, которые будут проверять данные перед записью в таблицу. В случае sequelize это можно сделать с помощью Hooks.

1. Разработка интерфейса взаимодействия пользователя с системой.

В интерфейсе данного веб-приложения 5 страниц:

1. Главная страница
2. Регистрация и авторизация
3. Личный кабинет администратора
4. Личный кабинет пользователя
5. Личный кабинет работника

Каждая страница состоит из отдельных компонентов, которые хранятся на клиенте в каталоге «components». Каждый компонент имеет свой функционал и назначение, это отражено в его названии. Например: «CarWashBar.js» служит для отображения панели с возможность выбрать филиал автомойки, благодаря этому компоненту пропадает нужда постоянно дублировать код при необходимости отображения панели филиалов на экране. Также в каталоге «components» есть подкаталог «modals», в котором хранятся модальные окна, эти окна отображаются поверх страниц при нажатии соответствующей им кнопке на странице. Их цель также убрать дублирование кода, например: «ChangePassword.js» написанный один раз, легко добавляется на страницы личных кабинетов и позволяет изменять свой пароль, как простым пользователям, так и администратору и работникам сети автомоек.

При разработке интерфейса страниц, их компонентов и модульных окон были использованы такие компоненты react-bootstrap как:

1. Modal
2. Button
3. Dropdown
4. ListGroup
5. ButtonGroup
6. Container
7. Form
8. Row
9. Col
10. Navbar
11. Nav
12. Table
13. OverlayTrigger
14. Tooltip
15. Card
16. Carousel
17. Spinner

Большинство элементов интерфейса обёрнуто в «observer» для автоматической перерисовки их в режиме реального времени без обновления страницы

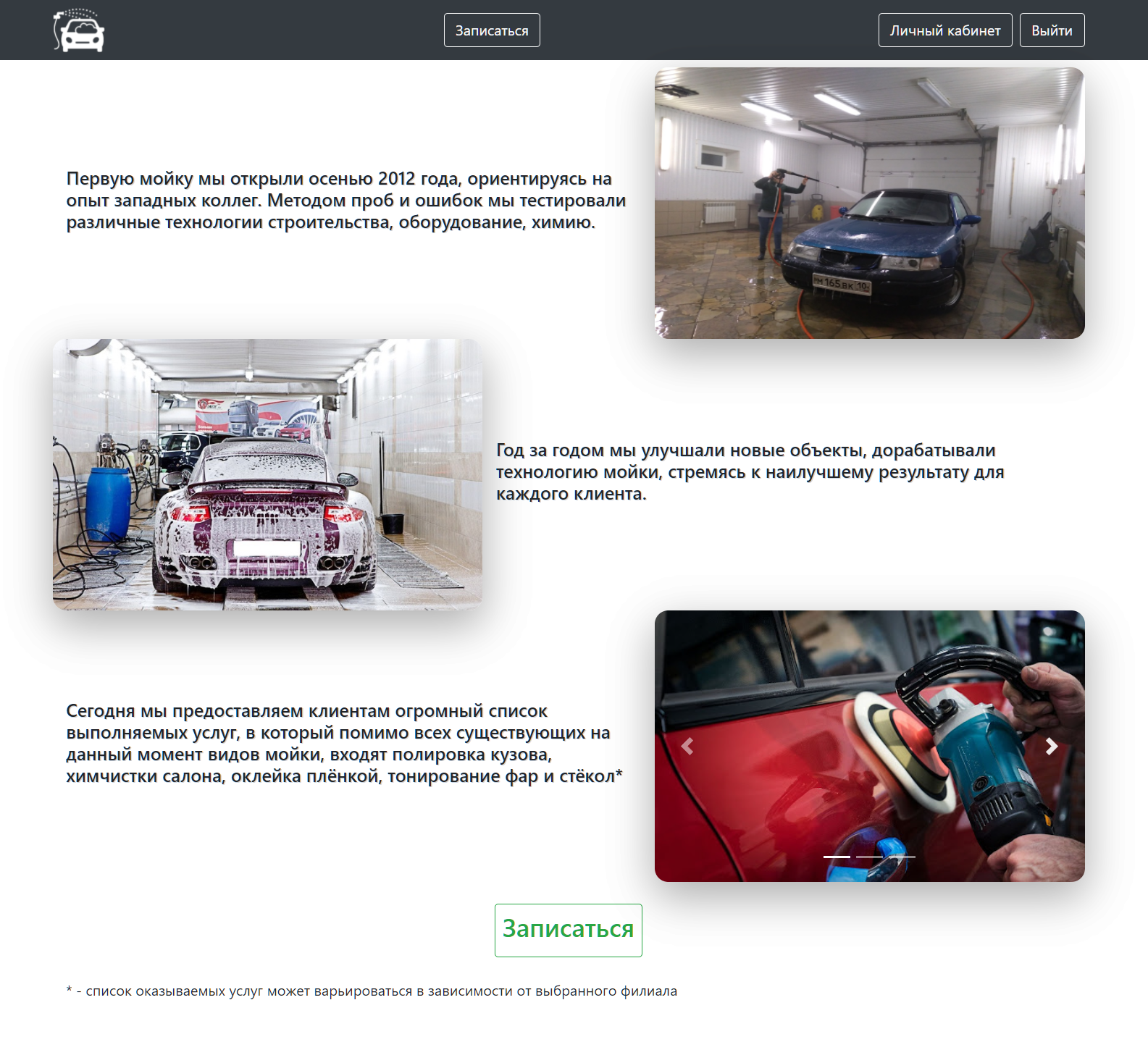


Рисунок 15. Главная страница веб-приложения.

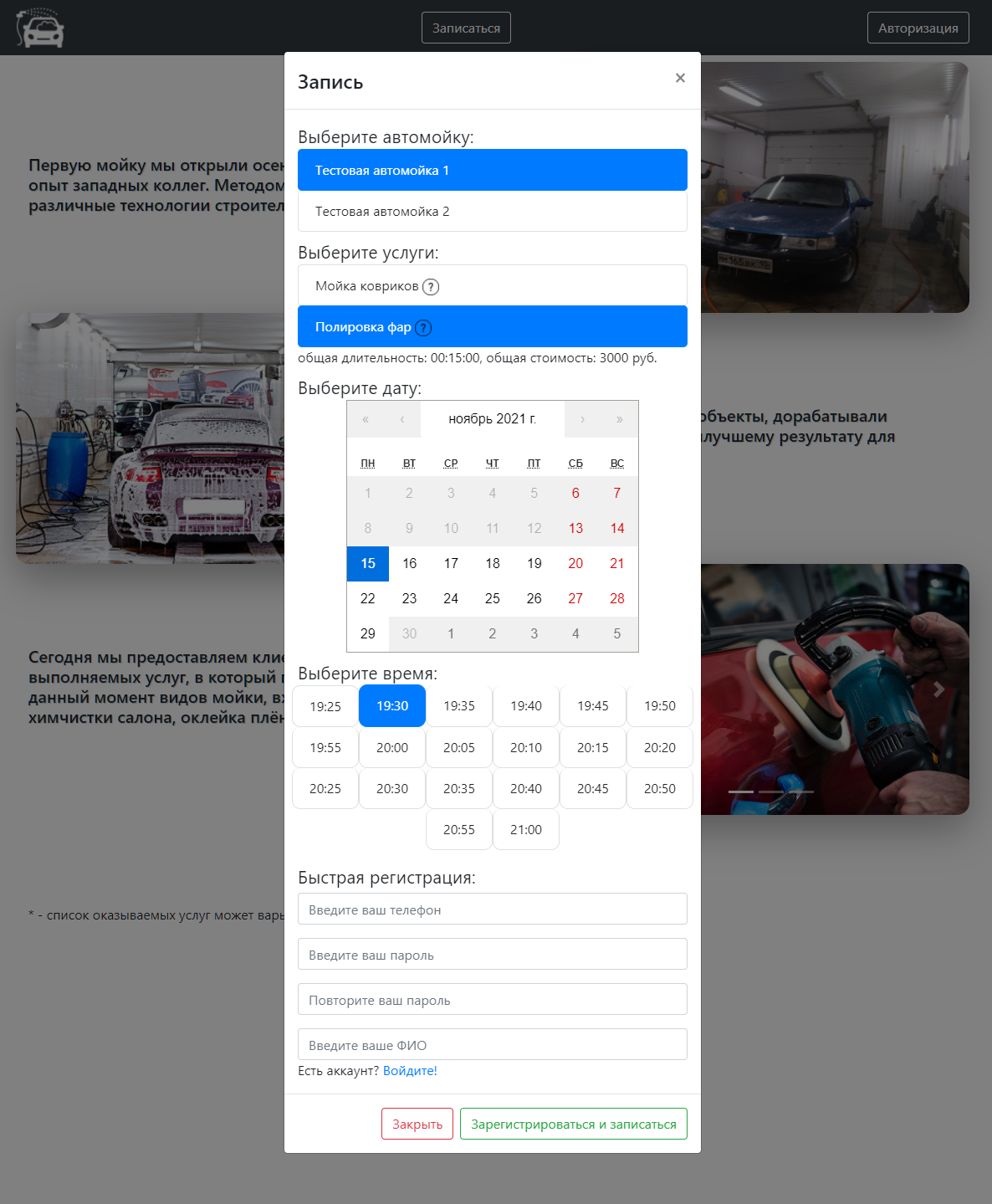


Рисунок 16. Модальное окно записи в автомойку не авторизованного пользователя.

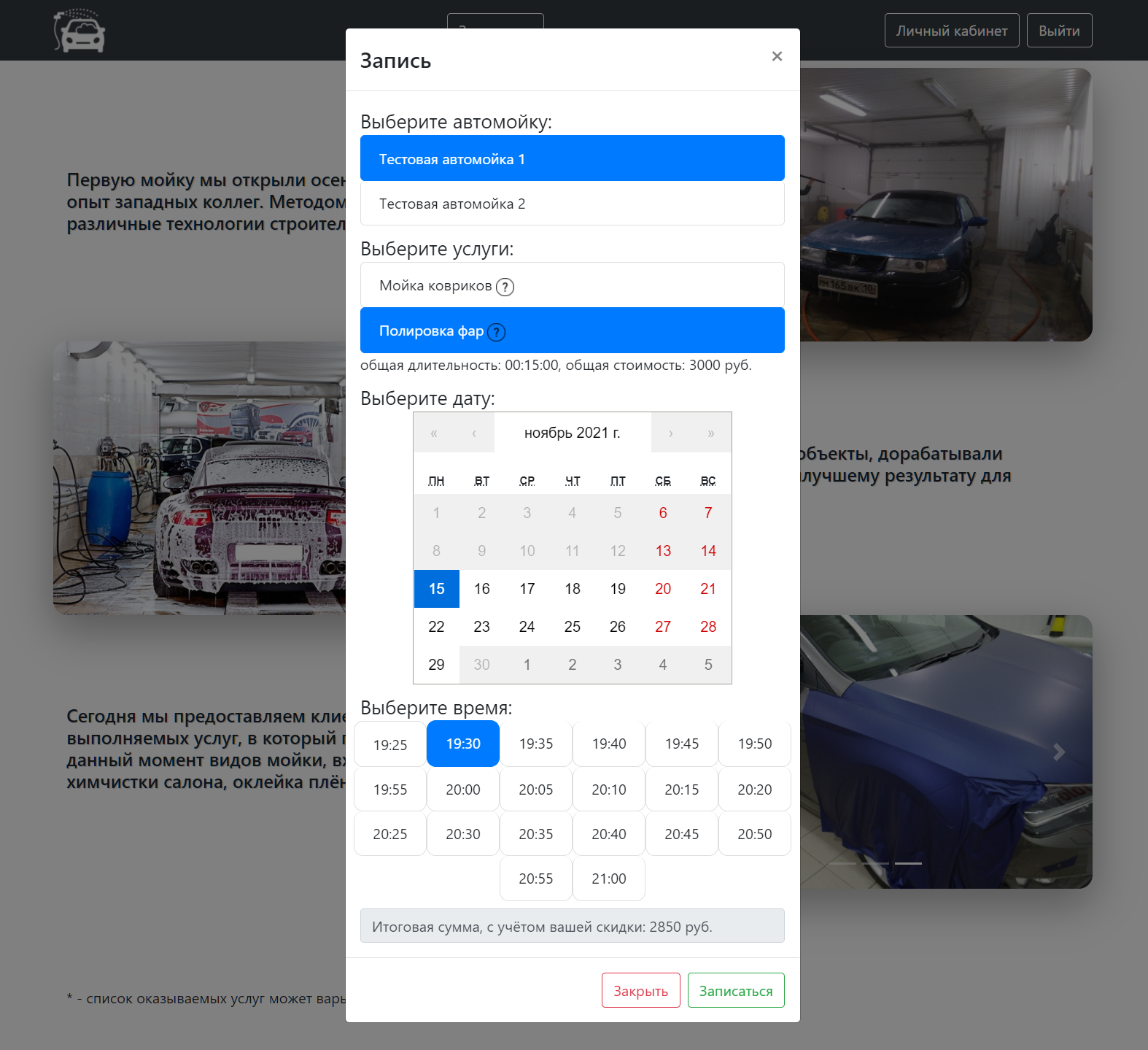


Рисунок 17. Модальное окно записи в автомойку если пользователь авторизовался.

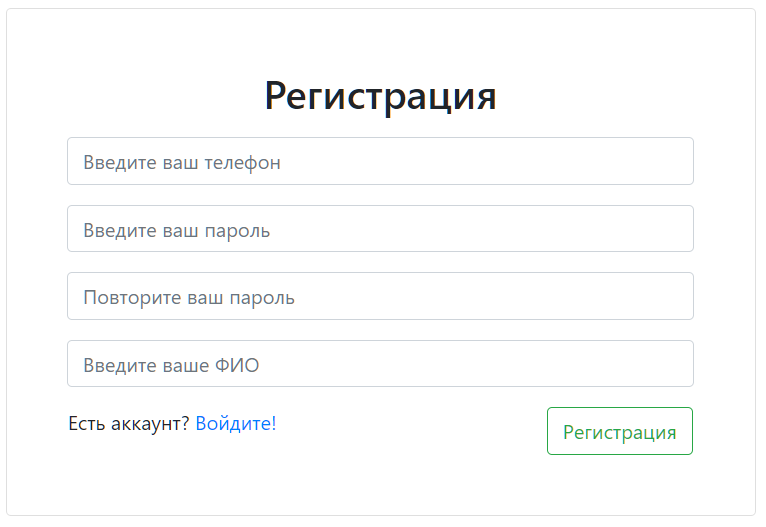
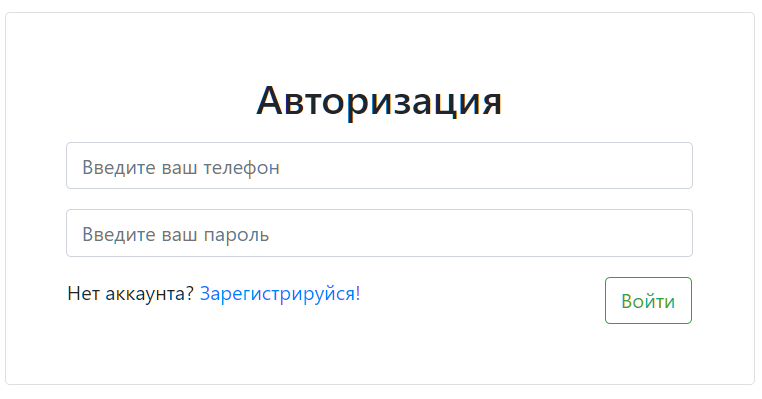


Рисунок 18. Формы регистрации и авторизации.

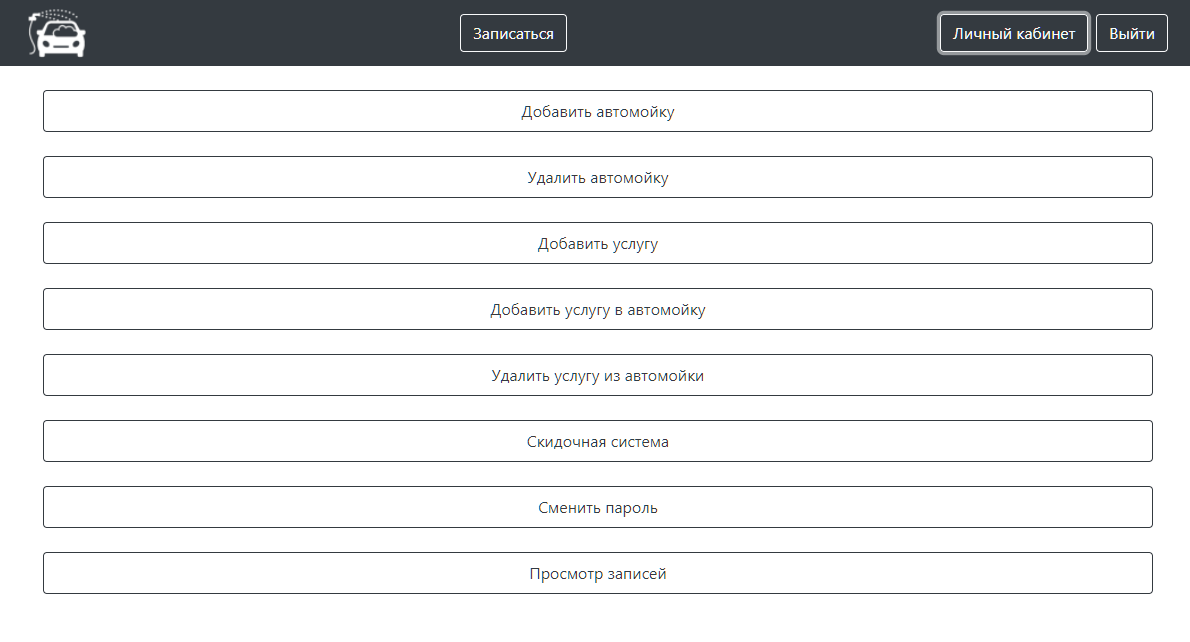
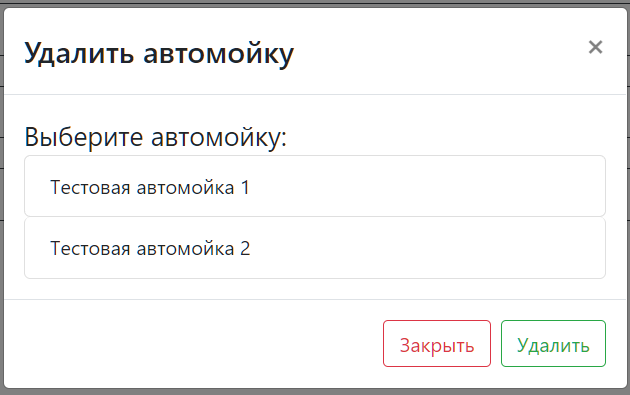
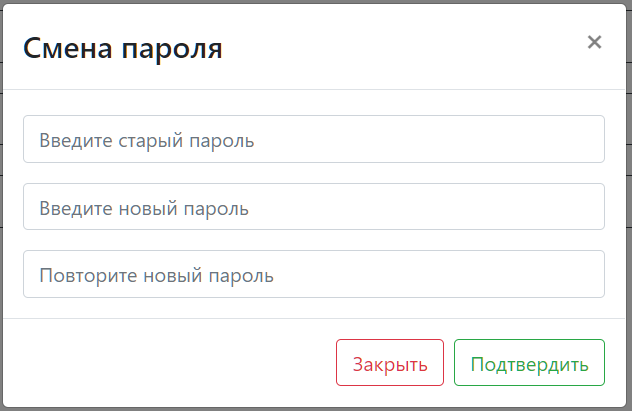
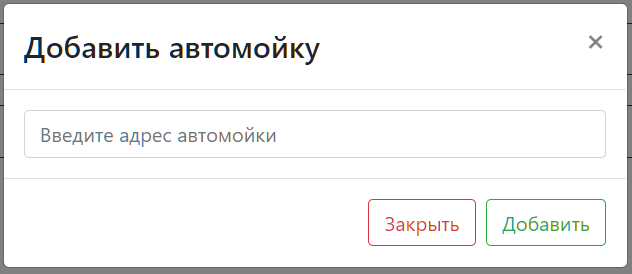


Рисунок 19. Личный кабинет администратора.



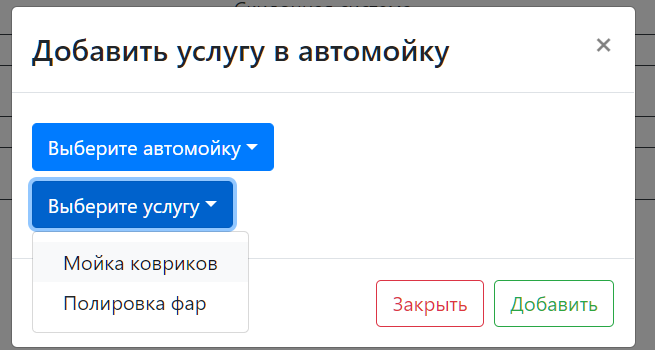
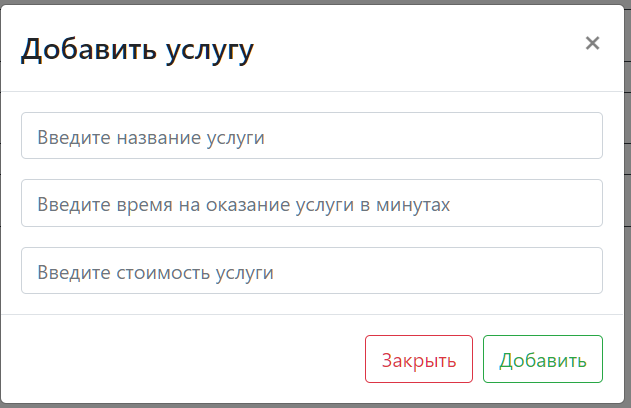
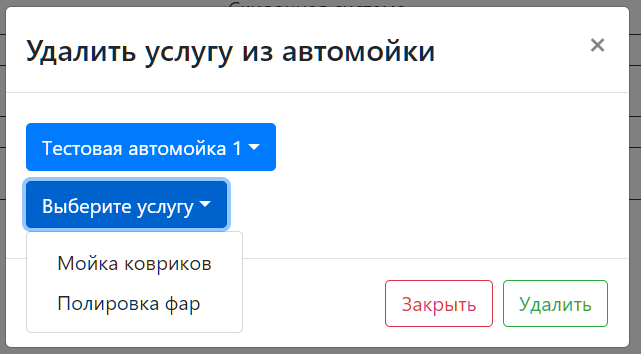
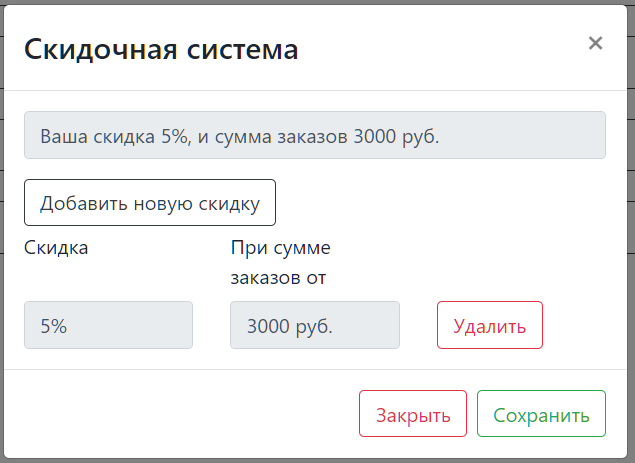
 

Рисунок 20. Все модальные окна из личного кабинета администратора.

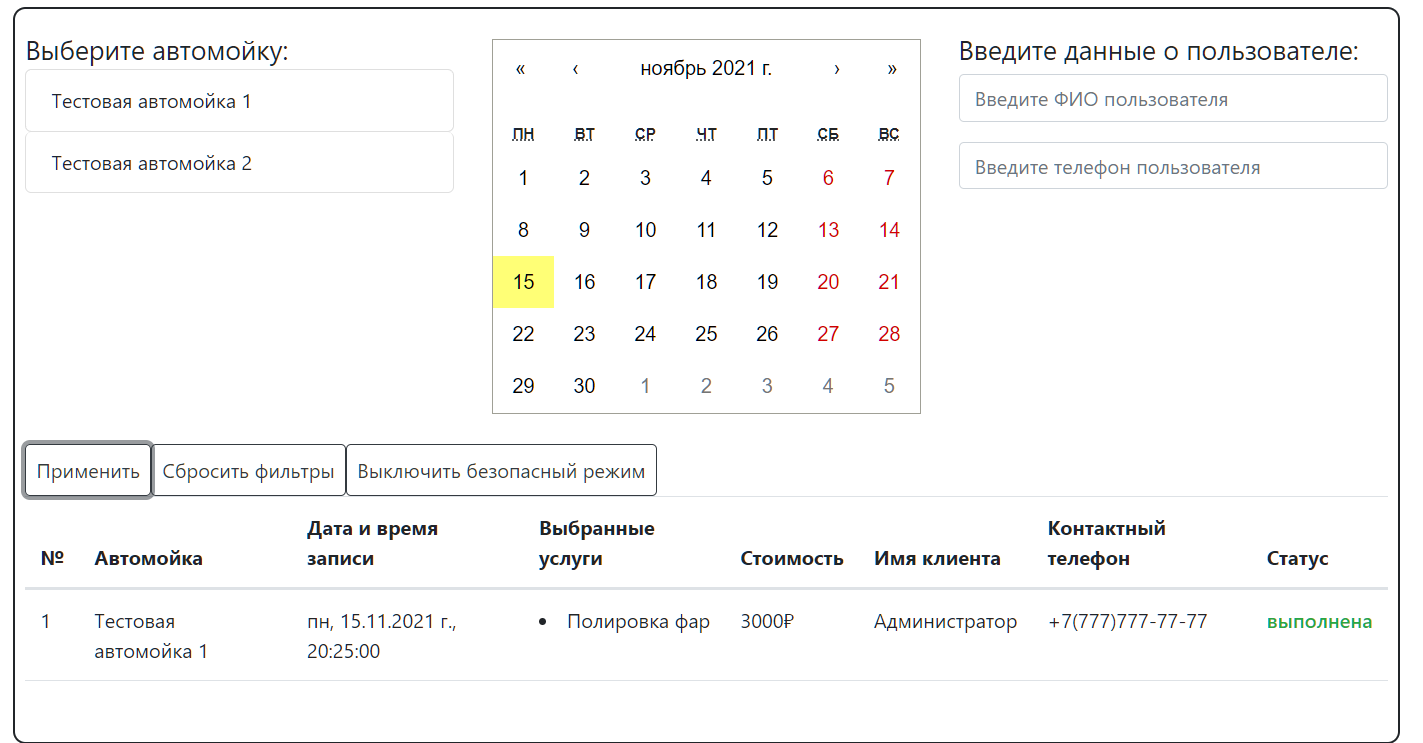


Рисунок 21. Просмотр всех записей с возможностью использования фильтров (нижняя кнопка в панели администратора).

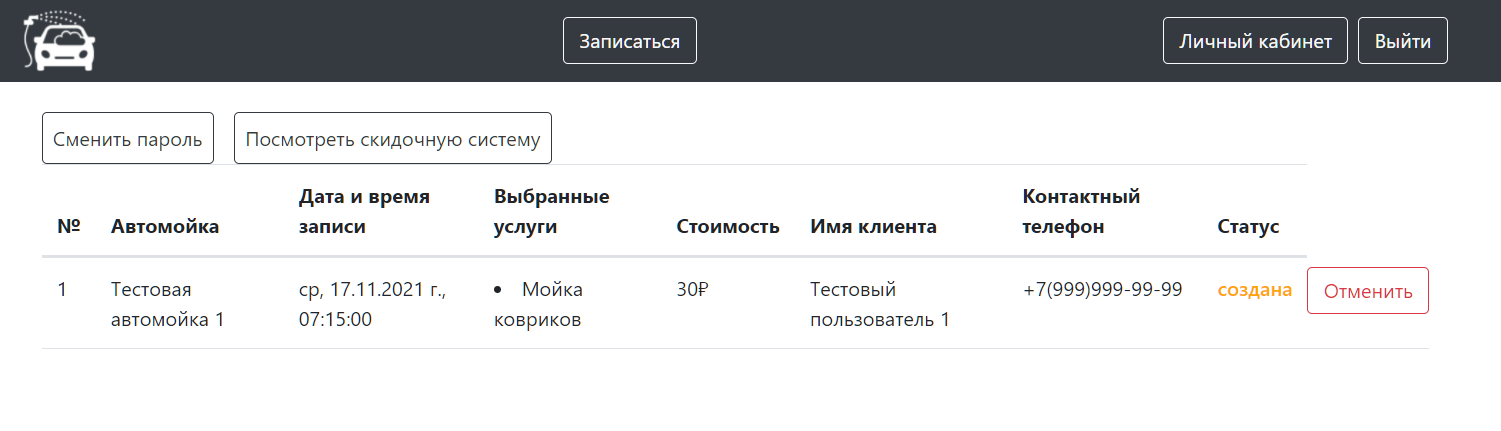


Рисунок 22. Личный кабинет пользователя.

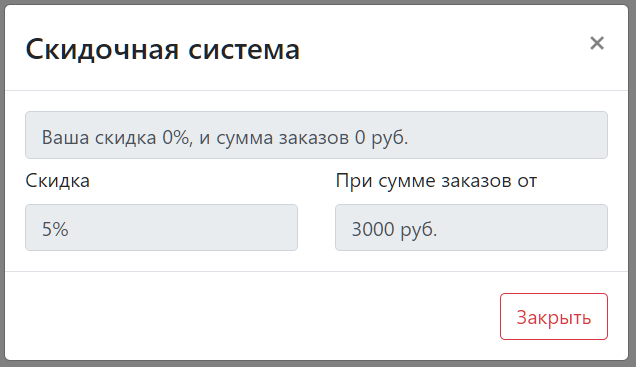


Рисунок 23. Вид скидочной системы для пользователя.

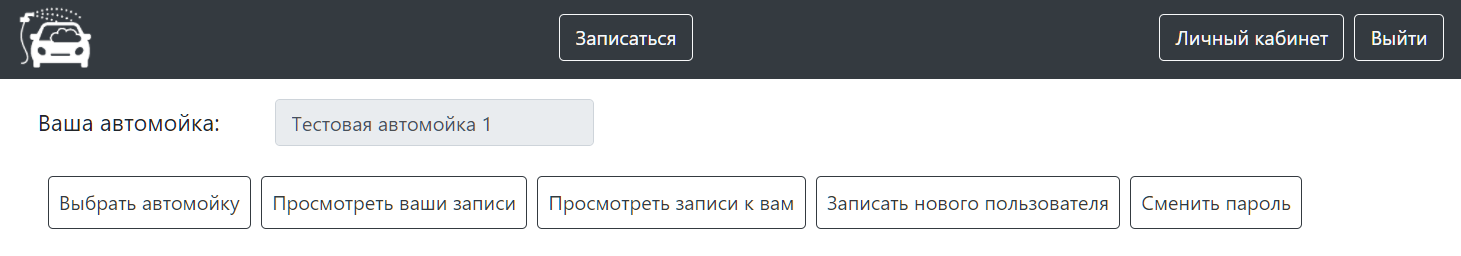


Рисунок 24. Личный кабинет работника.

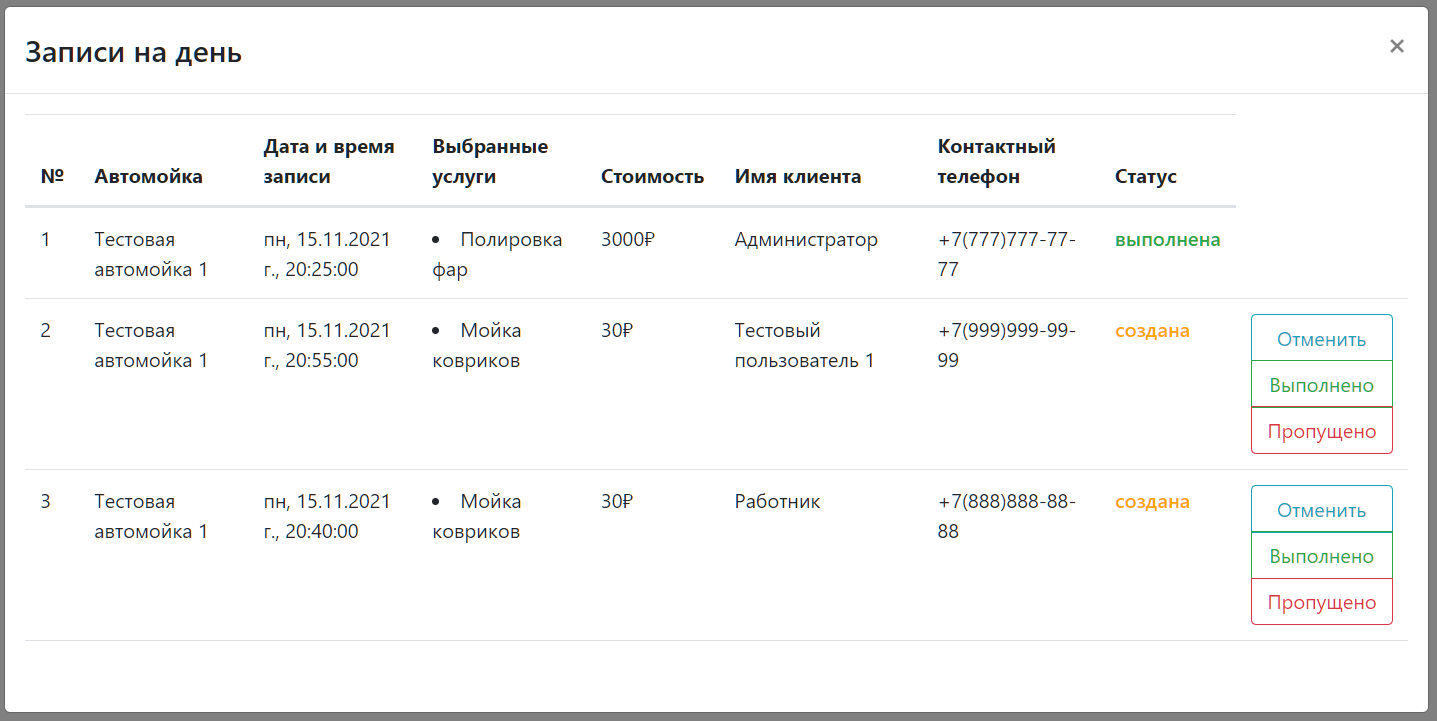


Рисунок 25. Просмотр работником записей на текущий день.

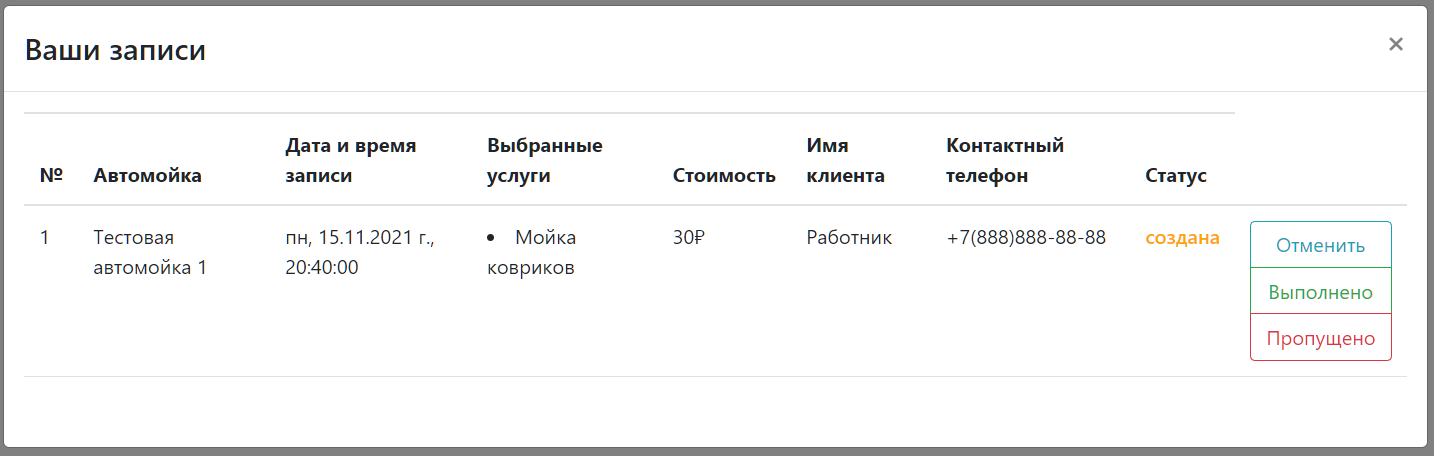


Рисунок 26. Просмотр работником всех своих записей.

1. Проектно-технологическая часть
2. Тестирование и отладка макета рабочей программы

Таблица 2. Тест-кейсы интерфейса.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Действие** | **Ожидаемый результат** | **Полученный результат** | **Примечание** |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Попытка записи без выбора филиала | Сообщение с просьбой выбрать филиал | Сообщение “Выберите автомойку” |  |
| 2 | Попытка записи без выбора услуг | Сообщение с просьбой выбрать услугу | Сообщение “Выберите услуги” |  |
| 3 | Попытка записи без выбора времени записи | Сообщение с просьбой выбрать время записи | Сообщение “Выберите время” |  |
| 4 | Попытка записи с выбором всех необходимых полей | Сообщение с информацией о успешной записи | Сообщение “Вы успешно записались” |  |
| 5 | Попытка авторизации на сайте с пустым или неверным паролем | Сообщение о неверном пароле | Сообщение “Указан неверный пароль” |  |
| 6 | Попытка авторизации на сайте с неправильным телефоном | Сообщение о том, что пользователя с таким телефоном не существует | Сообщение “Пользователь не найден” |  |
| 7 | Попытка регистрации без указания ФИО | Сообщение о незаполненном поле | Сообщение “Вы не ввели ФИО” |  |
| 8 | Попытка регистрации без ввода телефона и пароля | Сообщение о неправильности введённых данных | Сообщение “Некорректный телефон или password” | Текст сообщения исправлен |
| 9 | Повторение тест-кейса № 7 | Сообщение о неправильности введённых данных | Сообщение “Некорректный телефон или пароль” |  |

Продолжение таблицы 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 10 | Попытка регистрации с введением разных паролей | Сообщение о несовпадении паролей | Сообщение “Пароли не совпадают” |  |
| 11 | Попытка регистрации с указанием неполного номера телефона | Сообщение о некорректном номере телефона | Регистрация прошла успешно | Добавлена полная валидация номера телефона |
| 12 | Повторение тест-кейса № 10 | Сообщение о некорректном номере телефона | Сообщение “Некорректный телефон” |  |
| 13 | При авторизации указаны верные данные, вход под администратором | Успешная авторизация, появление кнопки «Личный кабинет» | Переход на главную страницу, появились кнопки «Личный кабинет» и «Выйти» |  |
| 14 | Нажать кнопку «Личный кабинет» | Личный кабинет администратора | Открылся личный кабинет администратора |  |
| 15 | В личном кабинете нажать кнопку «Добавить автомойку» | Появится модальное окно для ввода адреса автомойки |  |  |
| 16 | Нажать кнопку «Добавить», без заполнения поля | Сообщение о пустом поле | Добавление прошло успешно | Добавлена проверка на пустоту поля |
| 17 | Повторение тест-кейса № 15 | Сообщение о пустом поле | Сообщение “Вы ничего не ввели” |  |
| 18 | Нажать на кнопку «Закрыть» или на крестик в модальном окне или рядом с модальным окном | Окно добавления автомойки должно скрыться | Окно добавления автомойки скрывается при любом из перечисленных действий |  |

Продолжение таблицы 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 19 | В личном кабинете нажать кнопку «Удалить автомойку», выбрать автомойку и нажать кнопку «Удалить» | Автомойка удалится, и окно свернётся автоматически | Автомойка удалилась, произошёл переход на главную страницу | Убрана перезагрузка станицы и добавлено скрытие окна |
| 20 | Повторение тест-кейса № 18 | Автомойка удалится, и окно свернётся автоматически | Автомойка удалилась, и окно автоматически свернулось |  |
| 21 | В личном кабинете нажать кнопку «Добавить услугу», и нажать кнопку «Добавить», без заполнения полей | Сообщение о пустых полях | Ничего не произошло, окно осталось открытым. При не заполнении одного из полей результат аналогичный | Результат вполне приемлемый |
| 22 | В личном кабинете нажать кнопку «Добавить услугу», заполнить все поля, и нажать кнопку «Добавить» | Успешное добавление услуги | Успешное добавление услуги |  |
| 23 | В личном кабинете нажать кнопку «Добавить услугу в автомойку», и без выбора нажать кнопку «Добавить» | Ничего не произойдёт | Ничего не произошло |  |
| 24 | В личном кабинете нажать кнопку «Добавить услугу в автомойку», выбрать автомойку и услугу, и нажать кнопку «Добавить» | Услуга добавится в автомойку | Услуга добавилась в автомойку |  |
| 25 | В личном кабинете нажать кнопку «Добавить услугу в автомойку», и без выбора нажать кнопку «Удалить» | Ничего не произойдёт | Ничего не произошло |  |

Продолжение таблицы 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 26 | В личном кабинете нажать кнопку «Добавить услугу в автомойку», выбрать автомойку и услугу, и нажать кнопку «Удалить» | Услуга удалится из автомойки | Услуга удалилась из автомойки |  |
| 27 | В личном кабинете нажать кнопку «Скидочная система», «Добавить новую скидку» и нажать «Сохранить» | Сообщение, что не заполнены поля | Сообщение “Заполнены не все поля” |  |
| 28 | В личном кабинете нажать кнопку «Скидочная система», «Добавить новую скидку», заполнить все поля и нажать «Сохранить» | Скидка сохранится и будет отображаться в скидочной системе | Скидка сохранилась и отображается в скидочной системе |  |
| 29 | В личном кабинете нажать кнопку «Скидочная система», и нажать кнопу «Удалить» около любой строки | Скидка удалится | Окно свернулось, скидка удалилась и больше не отображается в списке |  |
| 30 | В личном кабинете нажать кнопку «Сменить пароль», ввести неверный старый пароль и нажать «Подтвердить» | Сообщение о неверном пароле | Сообщение “Указан неверный пароль” |  |
| 31 | В личном кабинете нажать кнопку «Сменить пароль», ввести разные новые пароли и нажать «Подтвердить» | Сообщение о разных паролях | Сообщение “Новые пароли не совпадают” |  |

Продолжение таблицы 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 32 | В личном кабинете нажать кнопку «Сменить пароль», не вводить новые пароли и нажать «Подтвердить» | Сообщение о некорректном новом пароле | Сообщение “Пароль не может быть пустым” |  |
| 33 | В личном кабинете нажать кнопку «Сменить пароль», ввести корректные данные и нажать «Подтвердить» | Сообщение об изменение пароля | Сообщение “Пароль успешно изменён” |  |
| 34 | В личном кабинете нажать кнопку «Просмотр записей» | Откроется режим просмотра всех записей | Режим открылся |  |
| 35 | В этом режиме выбрать в фильтрах филиал автомойки и нажать «Применить» | Отобразятся записи в выбранный филиал | Отобразились записи в выбранный филиал |  |
| 36 | Нажать кнопку «Сбросить фильтры» и затем «Применить» | Фильтры сбросятся, и затем отобразятся все записи в сеть автомоек | Фильтры сбросились, и затем отобразились все записи в сеть автомоек |  |
| 37 | Проверить работу кнопок изменения статуса для записей | Статус записи должен изменяться согласно кнопкам | Кнопки работают правильно, но после первого изменения статуса, повторное изменение не возможно |  |
| 38 | Нажать кнопку «Выключить безопасный режим» | Появится возможность редактировать статус любой записи | Появилась возможность редактировать статус любой записи по нескольку раз |  |
| 39 | Проверить работу фильтров по ФИО пользователя, телефону и дате | Отображение записей согласно фильтрам | Все фильтры работают корректно и дополняют друг друга |  |

Продолжение таблицы 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 40 | Нажать справа вверху кнопку «Выйти» | Выход из учётной записи | Выход из учётной записи |  |
| 41 | При авторизации указаны верные данные, вход под работником | Успешная авторизация, появление кнопки «Личный кабинет» | Переход на главную страницу, появились кнопки «Личный кабинет» и «Выйти» |  |
| 42 | Нажать кнопку «Личный кабинет» | Личный кабинет работника | Открылся личный кабинет работника |  |
| 43 | В личном кабинете нажать кнопку «Выбрать автомойку» выбрать другую и нажать «Подтвердить» | Выбранная автомойка изменится | Выбранная автомойка изменилась |  |
| 44 | В личном кабинете нажать кнопку «Просмотреть записи к вам» | Отобразится окно с записями на выбранную автомойку на текущий день | Окно отобразилось. Можно изменять статус записей, но только 1 раз |  |
| 45 | В личном кабинете нажать кнопку «Просмотреть ваши записи» | Отобразится окно с вашими записями на автомойку | Окно отобразилось. Можно изменять статус записей, но только 1 раз |  |
| 46 | В личном кабинете нажать кнопку «Записать нового пользователя» | Должно появится окно где совмещены запись и регистрация | Окно появилось |  |
| 47 | В появившемся окне корректно заполнить поля и нажать кнопку «Зарегистрировать и записать» | Сообщение с информацией о успешной записи | Сообщение “Вы успешно записались” |  |
| 48 | При авторизации указаны верные данные, вход под пользователем | Успешная авторизация, появление кнопки «Личный кабинет» | Переход на главную страницу, появились кнопки «Личный кабинет» и «Выйти» |  |

Продолжение таблицы 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 49 | Нажать кнопку «Личный кабинет» | Личный кабинет пользователя | Открылся личный кабинет пользователя |  |
| 50 | В личном кабинете нажать кнопку «Скидочная система» | Появится окно со списком скидок, скидкой пользователя и сумме услуг оплаченной пользователем | Окно появилось |  |

В результате тестирования были выявлены и устранены несколько ошибок.

1. Разработка руководства программиста и работника автомойки.

Руководство программиста: для запуска сервера в папке «server» нужно открыть окно PowerShell, нажав shift + правую кнопку мышки и выбрать соответствующий пункт. Затем написать команду “npm run dev” и нажать enter. Для остановки в этом же окне нажать ctrl+c, затем ввести “y” и нажать enter.

Для запуска клиентской части в папке «client» нужно открыть окно PowerShell, нажав shift + правую кнопку мышки и выбрать соответствующий пункт. Затем написать команду “npm start” и нажать enter. Для остановки в этом же окне нажать ctrl+c, затем ввести “y” и нажать enter.

Руководство для работника автомойки: при открытии веб-приложения перед вами будет главная страница с информацией о сети автомоек и предложением записаться. Для получения функционала работника необходимо в верхнем правом углу нажать кнопку «Авторизация» (Рисунок 27). Появится форма авторизации, в ней необходимо заполнить ввести данные вашей учётной записи (Рисунок 28) и нажать кнопку «Войти». После этого в правом верхнем углу появится кнопка «Личный кабинет», необходимо нажать её. Перед вами откроется личный кабинет работника (Рисунок 29).

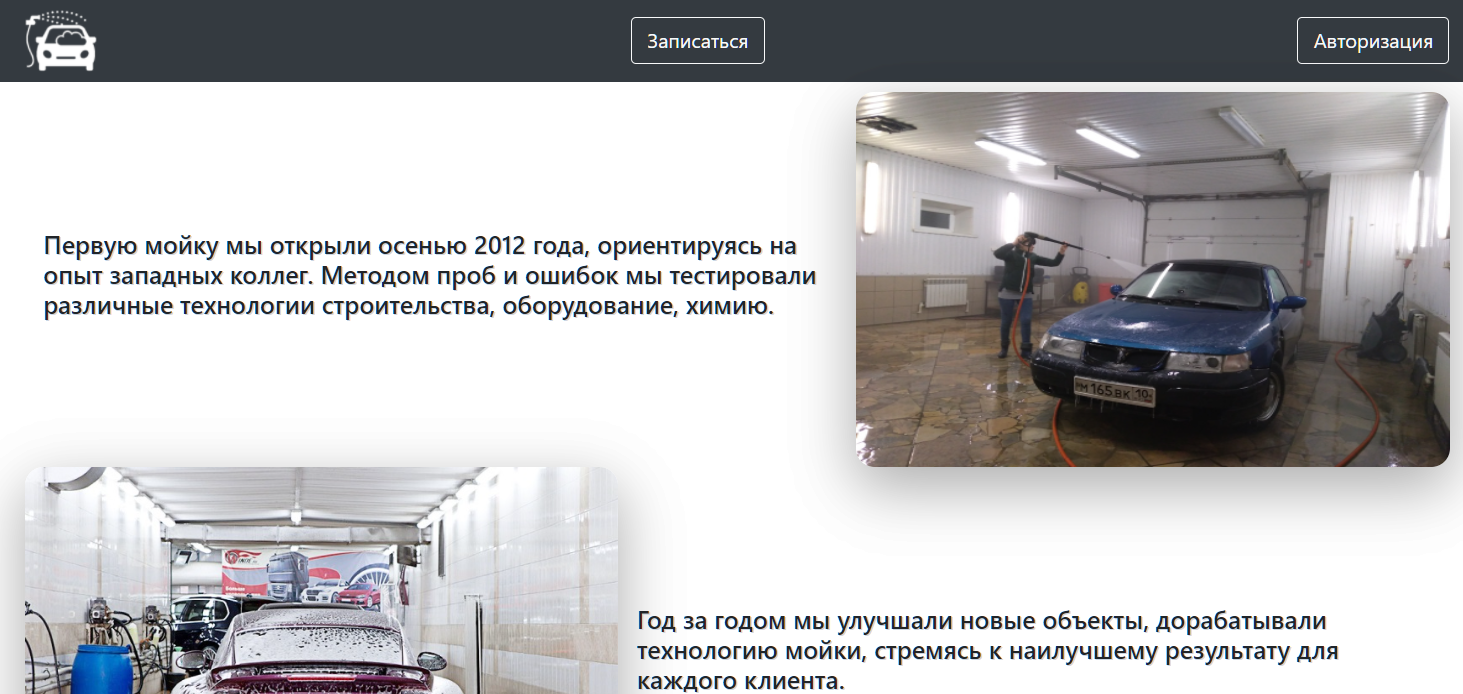
**

Рисунок 27. Главная страница сайта.

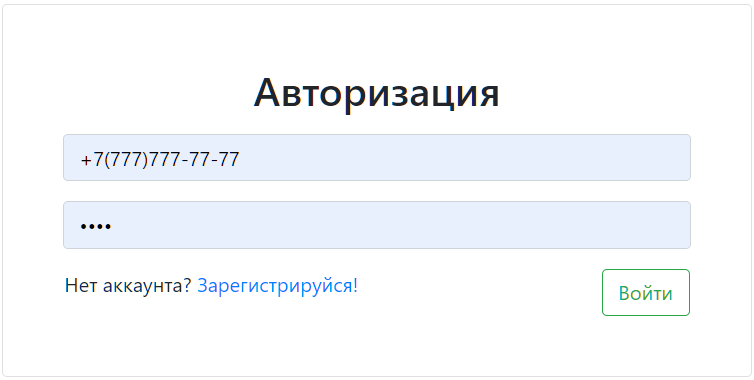
**

Рисунок 28. Форма авторизации.

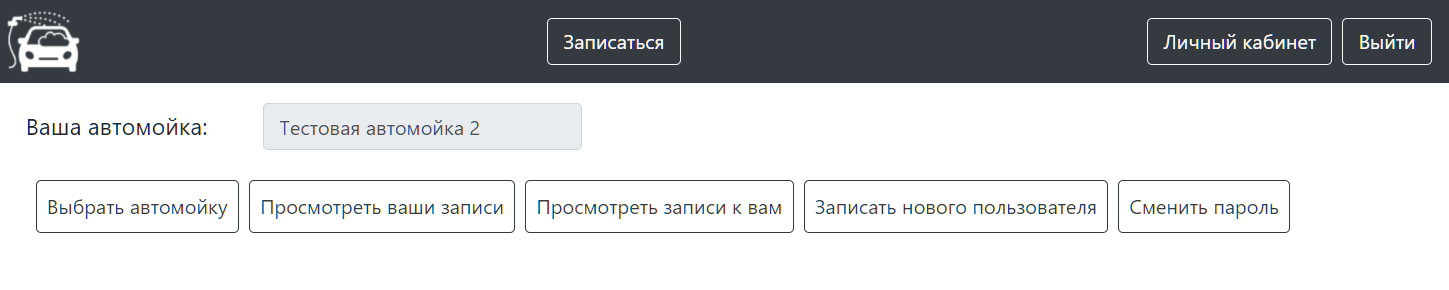
**

Рисунок 29. Личный кабинет работника.

Первым делом нужно выбрать филиал автомойки на котором предстоит сегодня работать, это можно сделать, нажав на кнопку «Выбрать автомойку» (Рисунок 30).

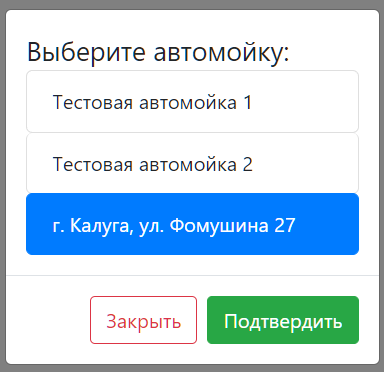
**

Рисунок 30. Личный кабинет работника.

Для просмотра записей на вашу автомойку на текущий день нужно нажать кнопку «Посмотреть записи к вам». В появившемся окне будет список записей с возможность изменения статуса (Рисунок 31): «Отменить», если клиент позвонил и попросил отменить запись; «Выполнено», если услуги оказаны и клиент заплатил деньги; «Пропущено», если клиент не явился в назначенное время.

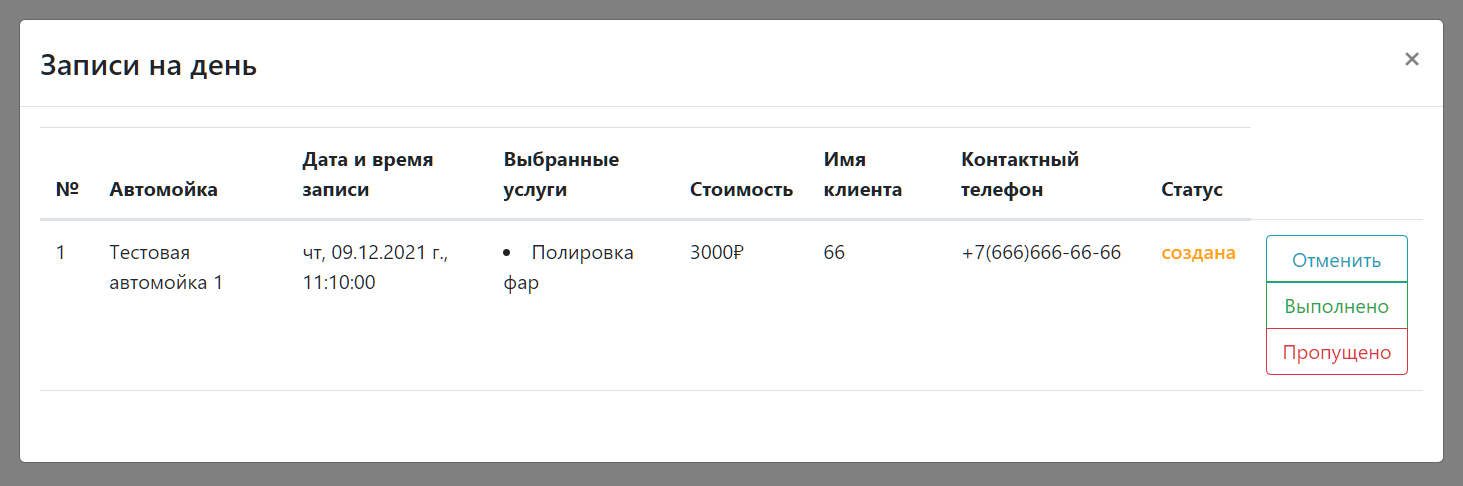


Рисунок 31. Просмотр записей работником автомойки.

Если пользователь впервые приехал на автомойку и хочет записаться, в лично кабинете есть кнопка «Записать нового пользователя». Появится окно записи (Рисунок 32) в котором нужно будет выбрать автомойку, услуги, дату и время по согласованию с клиентом. А также ввести его номер телефона, ФИО и пароль (который он сможет изменить в личном кабинете при необходимости), и нажать кнопку «Зарегистрировать и записать».

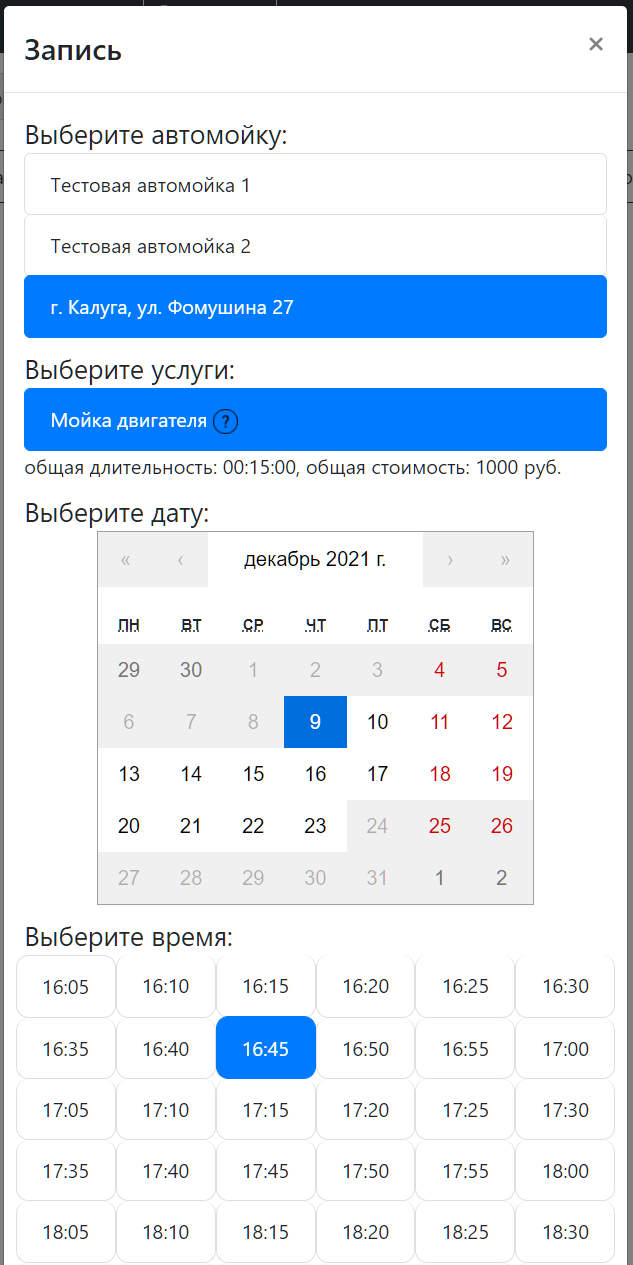
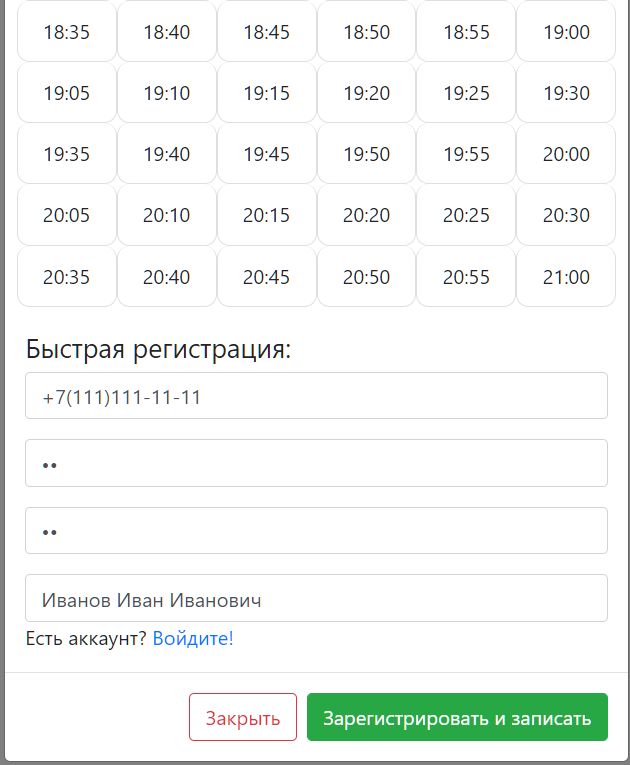
** **

Рисунок 32. Запись нового пользователя.

1. Вывод

В ходе написания курсовой работы были решены все поставленные цели и задачи. А именно, было написано веб-приложение для сети автомоек, которое позволило: избавиться от очередей, разгрузить менеджеров, сделать равномерную, полную загрузку автомоек и обеспечить удобство использования для клиентов.

Были закреплены навыки написания технической документации в соответствии с ГОСТами, которые необходимы для подготовки к выполнению выпускной квалификационной работы.

Также были сформированы практические навыки по разработке и реализации программного приложения на платформе Node.js с использованием библиотек React, React-Bootstrap и Sequelize. И были закреплены навыки работы и интеграции объектно-реляционной системы управления базами данных PostgreSQL.

1. Список использованных источников
2. ГОСТ 34.602-89 Техническое задание на создание АС. - Режим доступа: [https://www.mos.ru/dit/documents/tekhnicheskaia-politika/view/317220](https://www.mos.ru/dit/documents/tekhnicheskaia-politika/view/317220/)
3. Документация к front-end framework React-Bootstrap v2.0.3. - Режим доступа: [https://react-bootstrap.github.io/components](https://react-bootstrap.github.io/components/)
4. Документация к JavaScript-библиотеке для создания пользовательских интерфейсов React v17.0.2. - Режим доступа: <https://ru.reactjs.org/docs.html>
5. Документация к ORM-библиотеке Sequelize v6. - Режим доступа: [https://sequelize.org/master](https://sequelize.org/master/)
6. Документация к объектно-реляционной СУБД PostgreSQL v9.6.24. - Режим доступа: <https://postgrespro.ru/docs/postgresql/9.6/index>
7. Кириллов В.Ю. Методические указания по выполнению курсовой работы БАЗЫ ДАННЫХ. Калужский филиал МГТУ им. Н.Э. Баумана, 2018 г.
8. Коннолли, Бегг: Базы данных. Проектирование, реализация и сопровождение. Вильямс, 2017 г. – 1440с.
9. Сидоров В.Н., Сломинская Е.Н., Полникова Т.В., Макарова О.Ю. Оформление графической части выпускной квалификационной работы. Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2016.