**4. Проектно-технологическая часть**

* 1. **Проектирование начального и тестового наполнения базы данных. Порядок развертывания системы.**

Содержимое базы данных является основным источником информации для данного веб-приложения.

На странице регистрации с помощью специальной формы вручную можно добавить информацию о новом пользователе в базу данных, на странице добавления услуги — информацию о услуге, на странице уже существующей услуги — создать чат с исполнителем или оставить отзыв об услуге, а на странице своего профиля — изменить личные данные.

Формы, представленные на рисунках 4.1-4.5, позволяют создавать аккаунт, добавлять, изменять или удалять услуги, создавать чаты для коммуникации с исполнителями, добавлять комментарии (отзывы) об услугах без использования других сервисов работы с базой данных и помощи разработчиков.

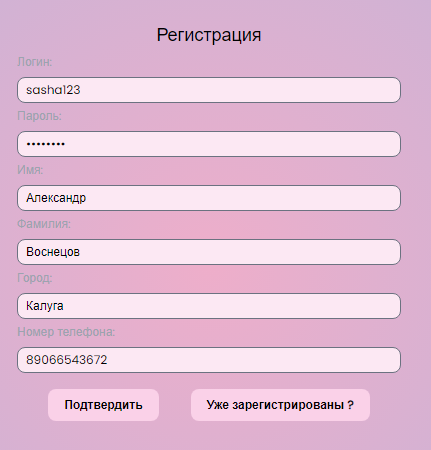
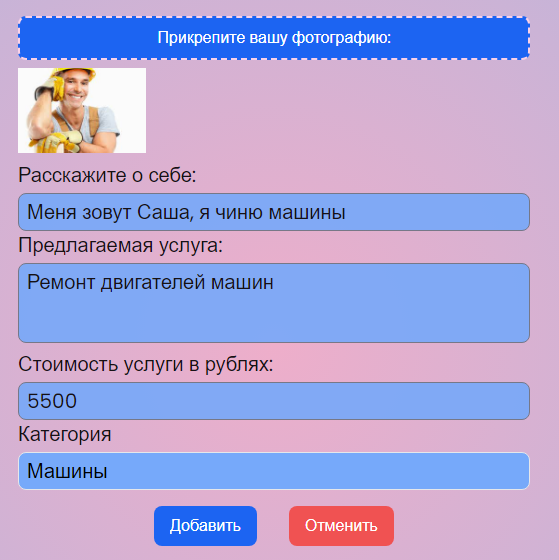


Рисунок 4.1 – Создание аккаунта



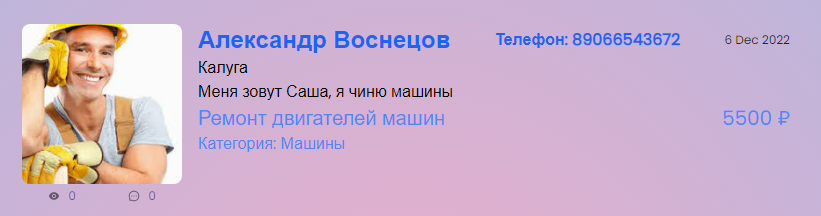


Рисунок 4.2 – Создание услуги

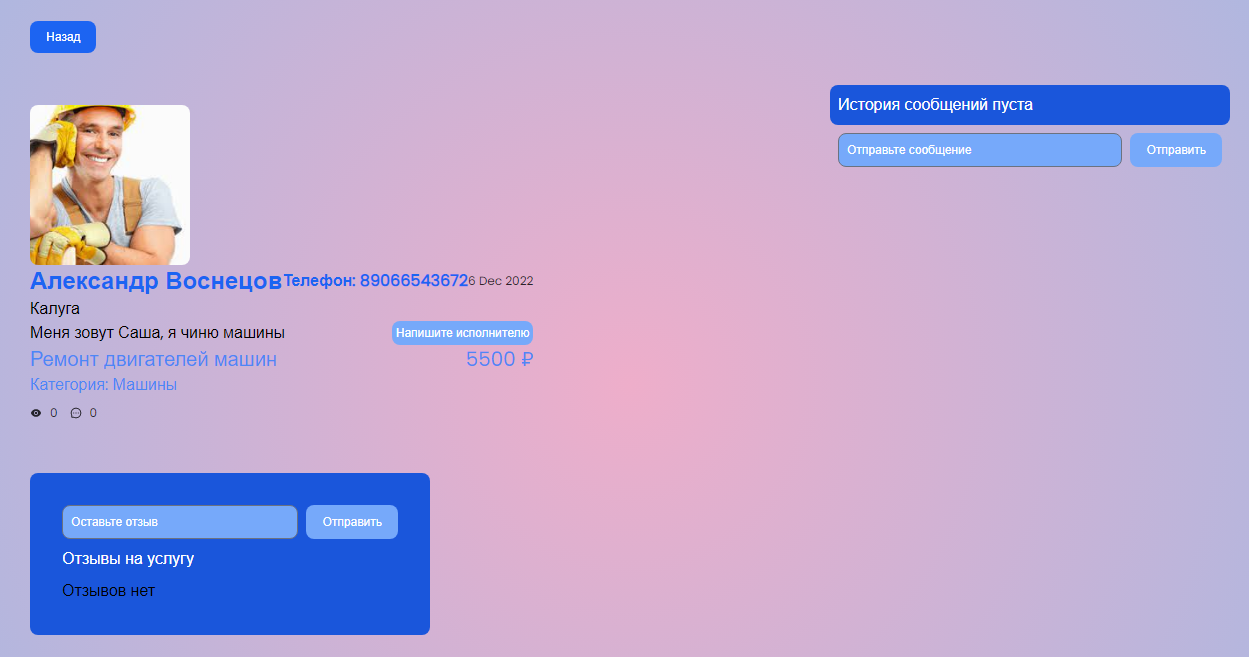


Рисунок 4.3 – Создание чата с исполнителем

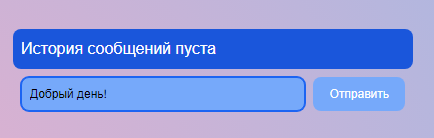
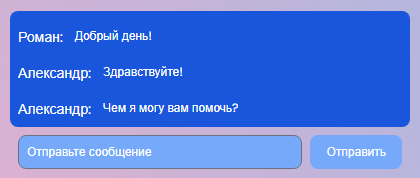
 

Рисунок 4.4 – Создание сообщений в чате

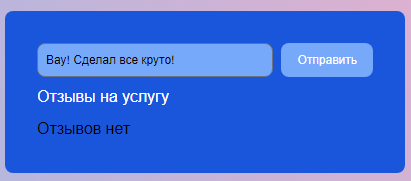
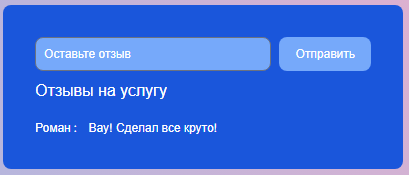
 

Рисунок 4.5 – Создание отзыва об услуге

Самая простая процедура развертывания сайта состоит из следующих этапов:

1. Перемещение файлов на сервер.
2. Запуск команды npm install для установки новых зависимостей.
3. Запуск node при помощи команды npm run dev. Также можно развернуть приложение на сервер PaaS-провайдера (Heroku).
4. Данная система тестировалась в следующих браузерах: Google Chrome, Firefox, Microsoft Edge.
   1. **Тестирование и отладка макета рабочей программы.**

Тестирование веб-приложения. Для начала необходимо запустить веб-приложение агрегатор услуг по ремонту. После запуска веб-приложения, открывается веб-страница с пользовательским интерфейсом (рисунок 4.6).

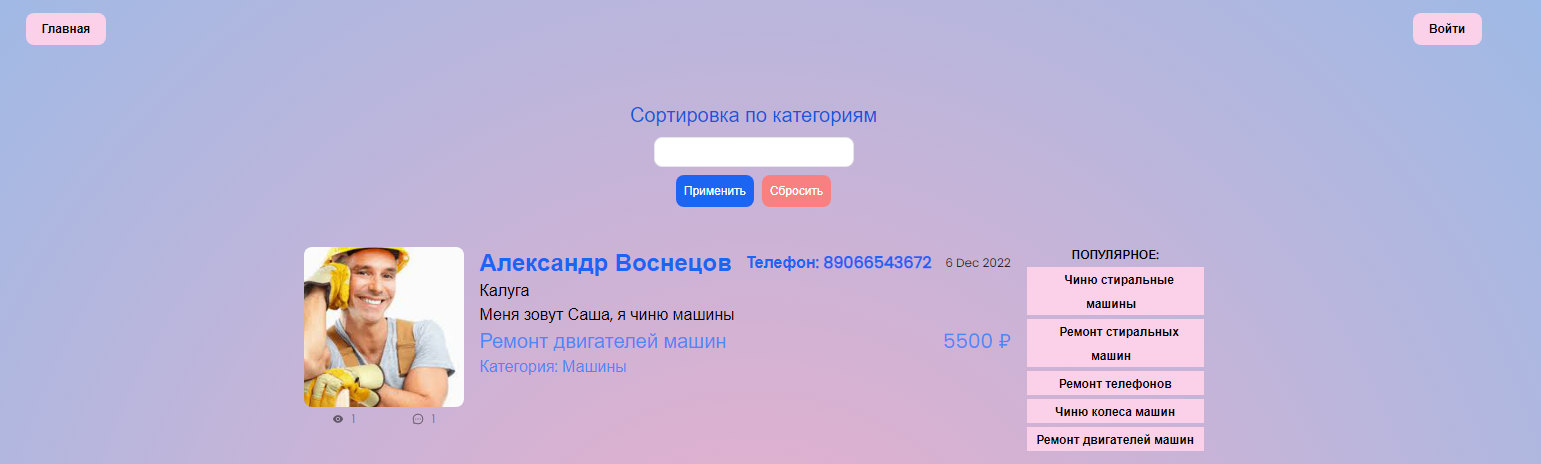
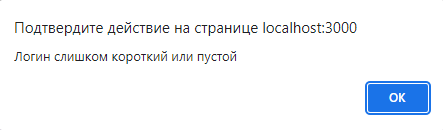


Рисунок 4.6 – Пример пользовательского интерфейса

Одной из самых первоначальных и главных задач веб-приложения является авторизация и регистрация пользователей. Для успешного выполнения этих задач, необходимо проверить правильность введенных данных. Например, длина пароля должна быть не менее 8 символов, а логина — не менее 3-ех. Для этого приведем пример ввода с ошибкой и проверим, как отреагирует на такую ситуацию веб-приложение (рис. 4.7).



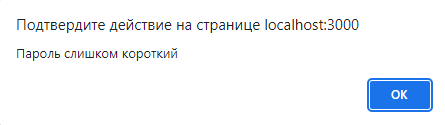


Рисунок 4.7 – Обработка ситуаций с некорректным вводом при регистрации

Следовательно, можно сделать вывод, что при создании пользователя, при вводе некорректных данных происходит успешная обработка исключительной ситуации.

Кроме того, при попытки входа под несуществующими данными, или при допущении ошибки в логине или пароле, приложение должно сообщить об этом пользователю, проверим это. Для этого введем несуществующий логин или пароль в нашей базе данных и посмотрим, что произойдет (рис. 4.8).

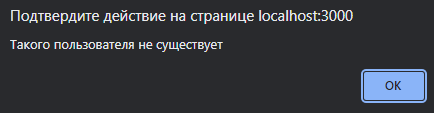


Рисунок 4.8 – Обработка несуществующего логина или пароля

Таким образом, веб-приложение выдает предупреждение о том, что пользователя с таким логином и паролем не существует.

Кроме того, может возникнуть ситуация, из-за которой соединение с базой данных не удастся произвести. Если нет соединения с базой данных – то корректная работа с сайтом невозможна. Следовательно, нужна проверка на успешное установление соединения с базой данных (рис. 4.9)

if (!services.length) {

        return (

            <div className='text-xl text-center text-white py-10'>

                Пока что нет услуг.

            </div>

        )

    }

Рисунок 4.9 – Проверка соединения с БД

В случае отсутствия соединения с БД, длина массива с услугами будет нулевой, так как от сервера не пришло ответа из-за отсутствия соединения с БД, поэтому дальнейшая работа с сайтом будет невозможна, однако пользователю будет показано, что на данный момент услуг нет (рис. 4.10).

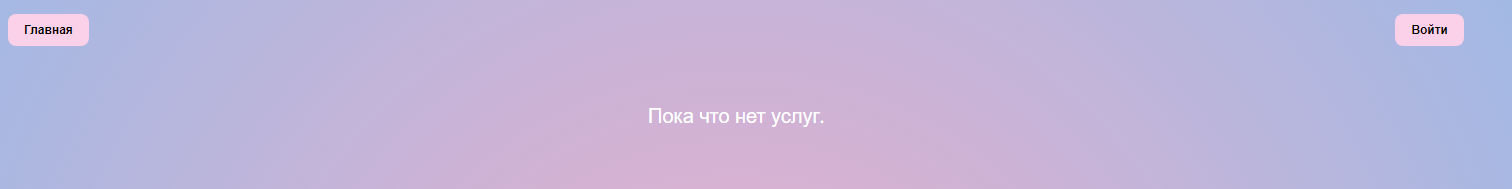


Рисунок 4.10 – Сообщение об ошибке

Так как ресурс является открытым, то появляется риск создания спаммерских или нецензурных комментариев под услугами. Чтобы этого избежать, используется фильтр в виде регулярного выражения, который будет блокировать создание нежелательных комментариев. Так, при попытке оставить нецензурный комментарий, пользователю отобразится предупреждение, и комментарий не будет создан (рис. 4.11).

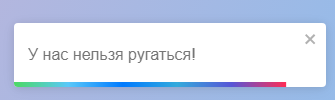


Рисунок 4.11 – Попытка добавления нецензурного отзыва

Основополагающую роль каждого веб-приложения играет пользовательский интерфейс. Его работа должна быть адаптивной под разную диагональ устройства, на котором будет открыто приложение (рис. 4.12).



Рисунок 4.12 – Пример адаптивности интерфейса под маленькое разрешение экрана

Таким образом, после проверки некорректной авторизации и регистрации, нежелательных комментариев, ситуации с отсутствием соединения с БД, а также проверки адаптивности под маленькое разрешение, можно сделать вывод, что веб-приложение работает корректно, а также осуществляет успешную обработку исключительных ситуаций.

**4.3. Разработка руководства пользователя**

Порядок работы:

После запуска открывается страница с пользовательским интерфейсом (рисунок 4.13).

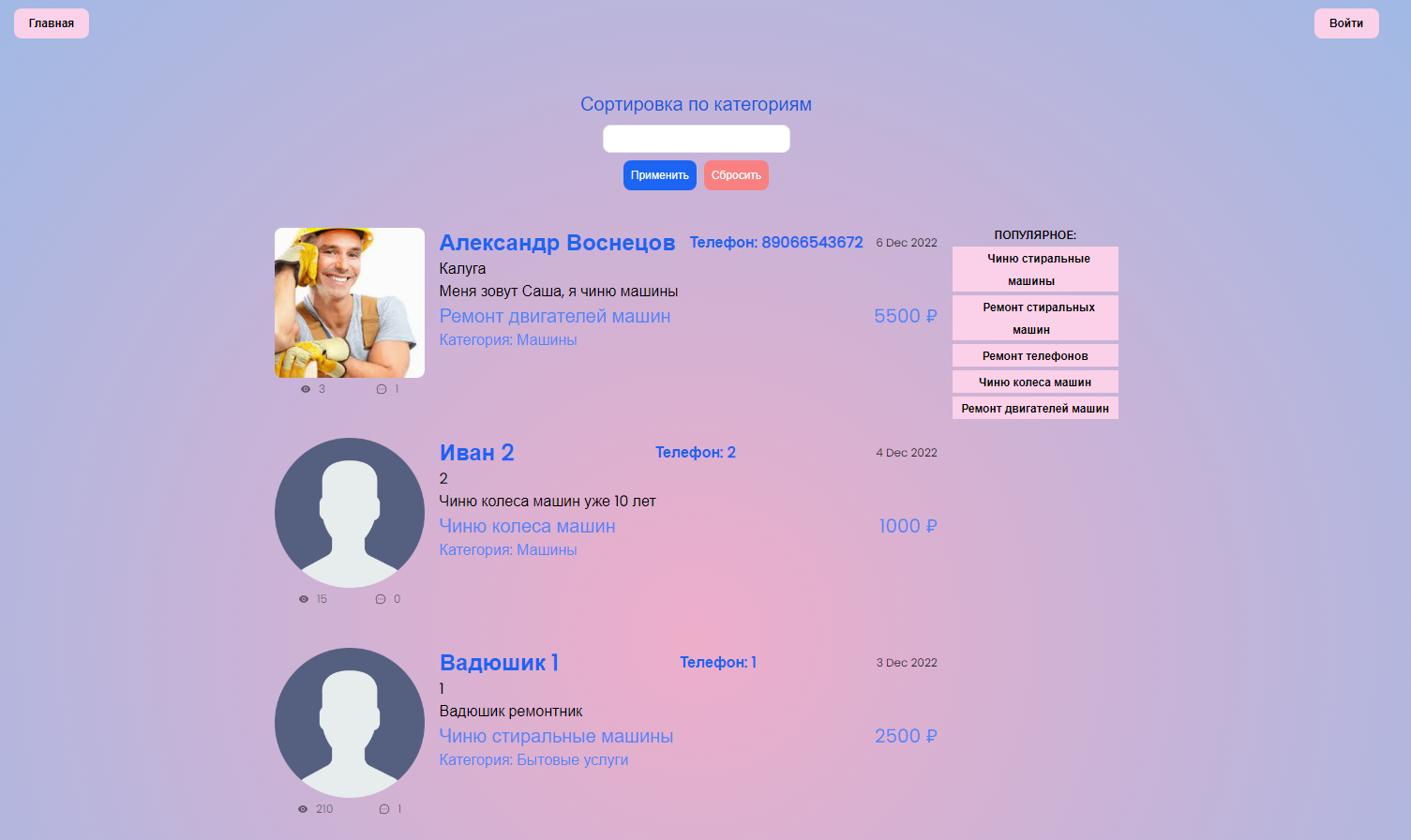


Рисунок 4.13 – Веб-приложение после открытия

Просматривать все существующие услуги на сайте (рис. 4.13), а также открыть конкретную услугу (рис. 4.14), может даже неавторизированный пользователь (гость).

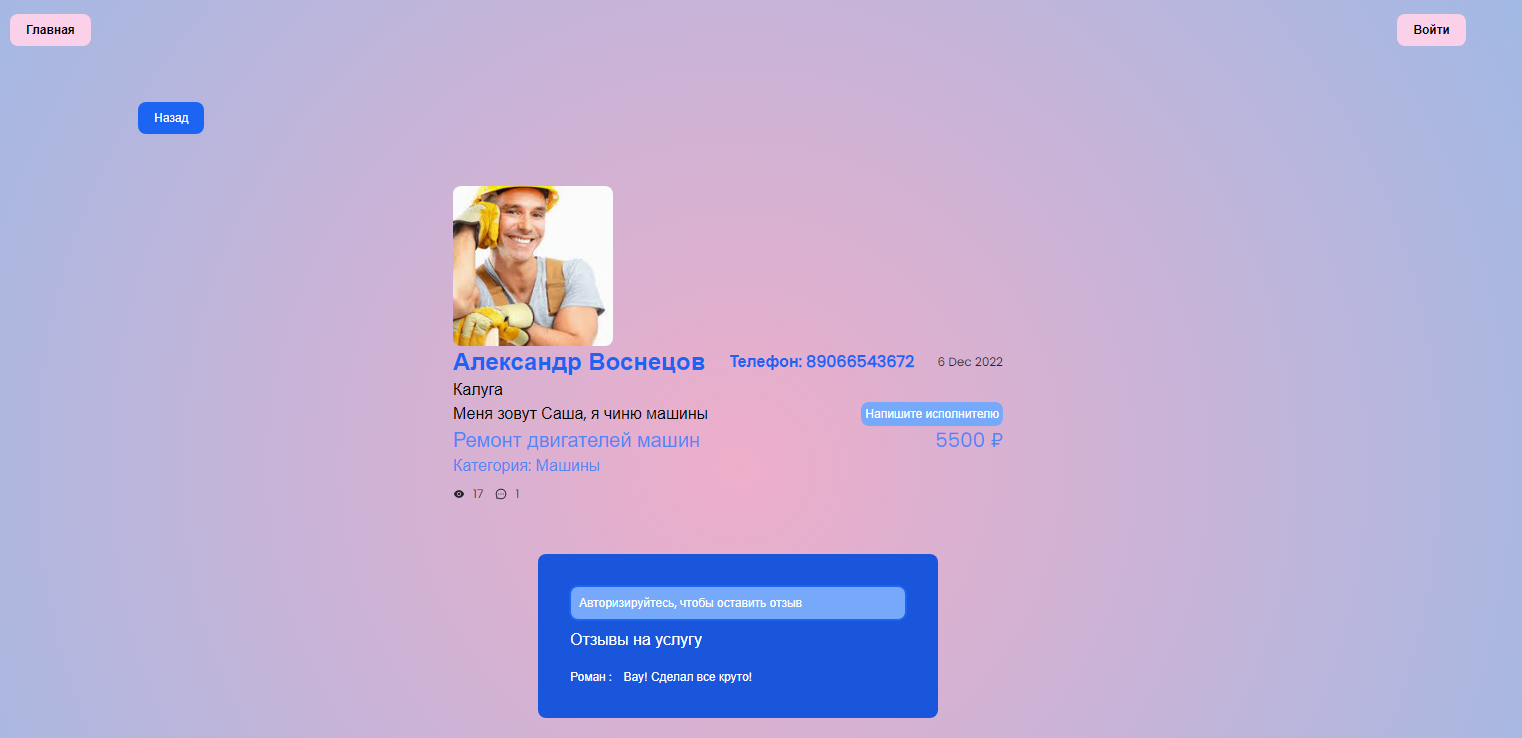


Рисунок 4.14 – Страница с услугой у гостя

Далее следует разделить пользователей данного сайта на 2 категории: исполнитель и заказчик. Однако это разделение является условным, так как абсолютно любой пользователь может быть как исполнителем, так и заказчиком услуги.

Чтобы появилась возможность связываться с исполнителями, а также добавлять услуги, необходимо пройти процессы регистрации и авторизации. При входе в систему пользователю надо ввести логин и пароль (Рисунок 4.15). Если ранее пользователь не был зарегистрирован, он может заполнить соответствующую форму для регистрации (Рисунок 4.16). При некорректном вводе данных пользователю выводится сообщение об ошибке (Рисунок 4.17). Например, указан неправильный логин (менее 3-ех символов) или пароль (менее 8 символов) или при входе указан незарегистрированный логин или неправильный пароль. Сообщение об ошибке появляется в правом нижнем углу.



Рисунок 4.15 - Форма авторизации



Рисунок 4.16 - Форма регистрации

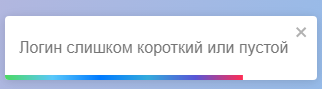
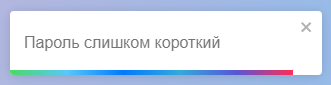
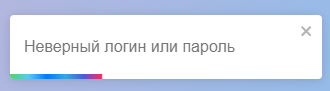
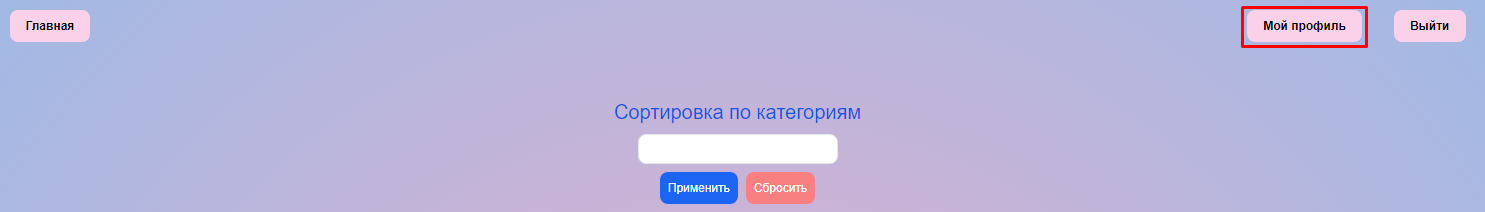
  

Рисунок 4.17 – Сообщения об ошибке при авторизации или регистрации

После регистрации и авторизации исполнитель может добавить услугу. Для этого надо нажать на кнопку «Мой профиль», а затем «Добавить услугу» (рис. 4.18).



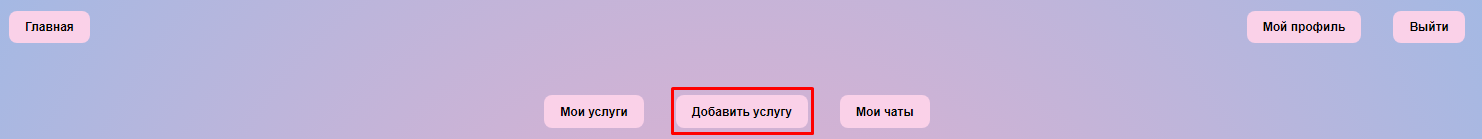


Рисунок 4.18 – Путь для добавления услуги

После этого исполнитель попадет на страницу добавления услуги (рис. 4.19).

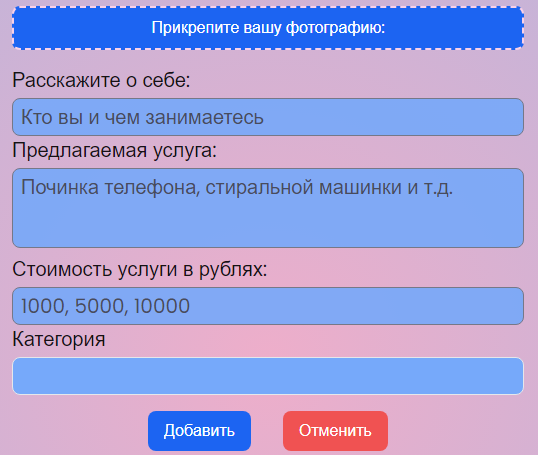
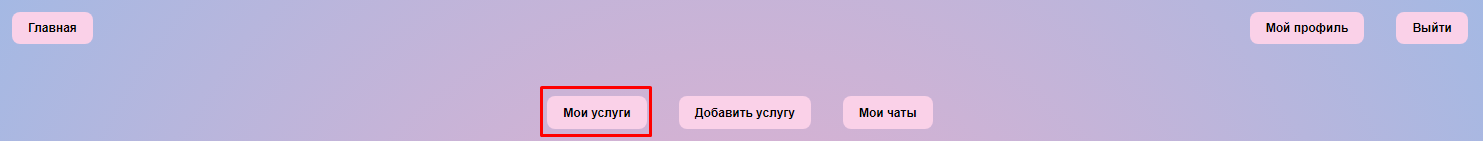


Рисунок 4.19 – Форма для добавления услуги

Кроме того, он может просмотреть все свои услуги (4.20).



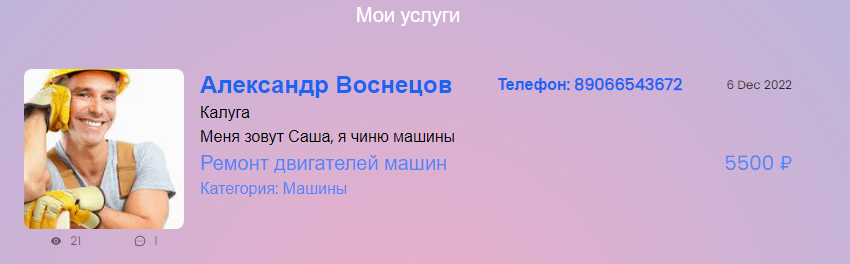


Рисунок 4.20 – Просмотр всех услуг исполнителя

Если исполнитель перейдет на свою конкретную услугу, нажав на нее, то откроется страница данной услуги и отзывы к ней, а также появится возможность редактирования или удаления услуги (рис. 4.21).

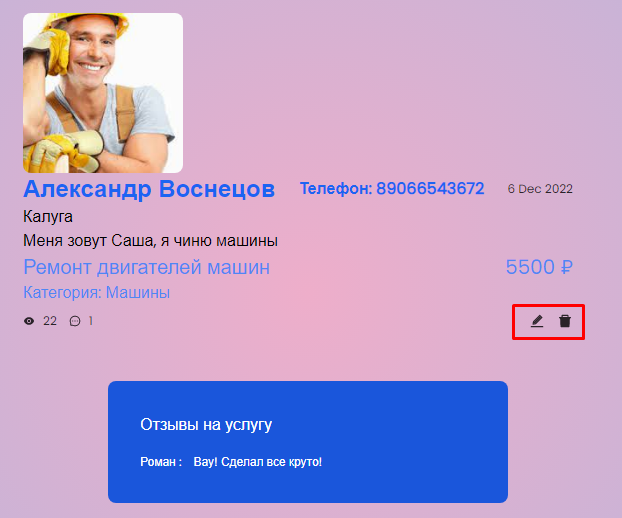


Рисунок 4.21 – Возможность редактирования или удаления своей услуги

Заказчик, в свою очередь, перейдя на конкретную услугу, может создать чат с исполнителем или оставить отзыв к ней (рис. 4.22).

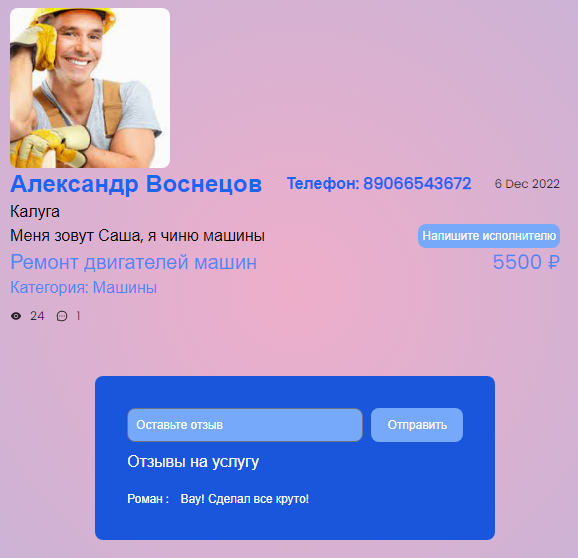


Рисунок 4.22 – Возможность создания чата с исполнителем или добавления отзыва об услуге

При нажатии на кнопку «Напишите исполнителю» создастся чат, там можно общаться и договариваться по поводу конкретной услуги (4.23)



Рисунок 4.23 – Создание чата с исполнителем

И исполнитель, и заказчик могут посмотреть все свои чаты на странице своего профиля, нажав на кнопку «Мои чаты» (рис. 4.24). Кроме того, на этой странице можно изменять информацию о пользователе (имя, фамилию, город и номер телефона).

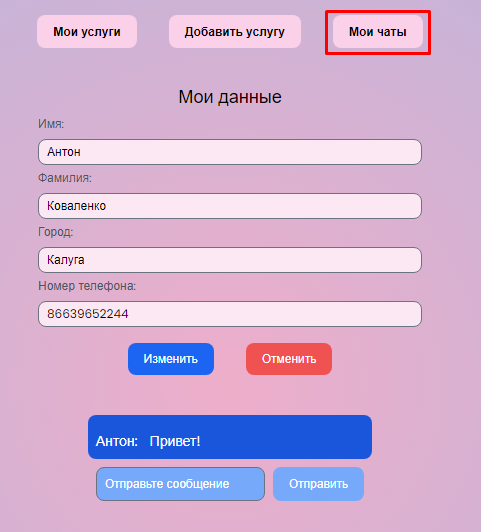


Рисунок 4.24 – Отображение всех чатов пользователя и возможность изменения данных пользователя

Администратор (модератор) может удалять, изменять любые услуги, а также удалять отзывы к ним (рис. 4.25).

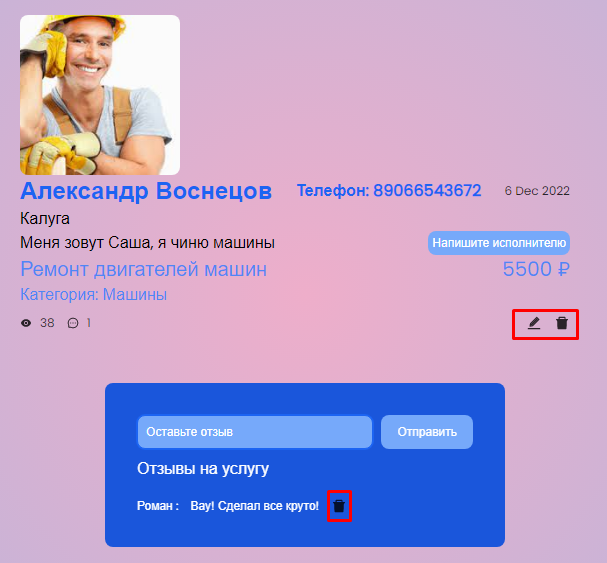


Рисунок 4.25 – Возможность удаления, изменения услуг и удаления отзывов у администратора

**Заключение**

При выполнении курсовой работы на тему «веб-приложение агрегатор услуг по ремонту» была исследована и описана предметная область, проведен анализ аналогов данной системы, а также выбор инструментов и платформы для разработки.

Проведен анализ объектов автоматизации и разработаны методы решения технических задач. Также была разработана и реализована структура базы данных, архитектура приложения, серверная часть приложения и веб-интерфейс системы. База данных была наполнена тестовыми данными.

Результатом данной курсовой работы является рабочий сайт-агрегатор услуг по ремонту.

**Список использованных источников**

1. Арно Лоре Проектирование веб-API / Пер. с англ. Д. А. Беликова. – М.: ДМК Пресс, 2020.– 440 с

2. Карпова, И.П. Базы данных: Учебное пособие / И.П. Карпова. - СПб.: Питер, 2013. - 240 c.

3. Янг А., Мек Б., Кантелон М. Я Node.js в действии. 2-е изд. — СПб.: Питер, 2018. — 432 с.: ил. — (Серия «Для профессионалов»).

4. https://mongoosejs.com/docs/queries.html

5. Ревунков Г.И., Ковалева Н. А., Силантьева Е. Ю. Проектирование баз данных. [Электронный ресурс] – МГТУ им. Н.Э.Баумана, 2018. – 48 с. Режим доступа: <https://e.lanbook.com/reader/book/103499/>

6. <https://nodeguide.ru>

7. Руководство по программированию на JavaScript [Электронный ресурс] – режим доступа: https://developer.mozilla.org/ru/docs/Learn/Server-side/Express\_Nodejs/skeleton\_website

8. Изучаем Node.js [Электронный ресурс] – режим доступа: https://metanit.com/web/nodejs/7.1.php 8. Learning JavaScript Design Patterns — Addy Osmani 2020 – 439c

9. https://reactdev.ru/libs/redux-toolkit/