**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

**АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ**

**ИМ. Б.Л. РОЗИНГА (ФИЛИАЛ) СПбГУТ**

**(АКТ (ф) СПбГУТ)**

**КУРСОВОЙ ПРОЕКТ**

### НА ТЕМУ

|  |  |
| --- | --- |
| РАЗРАБОТКА ПОДСИСТЕМЫ ОФОРМЛЕНИЯ | |
|  | ЗАКАЗОВ У РЫБОЛОВОВ |
|  | Л109. 23КП01. 010 ПЗ |

(Обозначение документа)

|  |
| --- |
| МДК.02.01 Технология разработки |
| программного обеспечения |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИСПП-01 | | 06.12.2023 | А.И. Кузьмин |
|  | (Группа) | (Подпись) | (Дата) | (И.О. Фамилия) |
| Преподаватель |  |  | 06.12.2023 | Ю.С. Маломан |
|  |  | (Подпись) | (Дата) | (И.О. Фамилия) |

Архангельск 2023

**СОДЕРЖАНИЕ**

Перечень сокращений и обозначений 3

Введение 4

1 Анализ и разработка требований 6

1.1 Назначение и область применения 6

1.2 Постановка задачи 6

1.3 Описание алгоритма функционирования системы 6

1.4 Выбор состава программных и технических средств 8

2 Проектирование программного обеспечения 10

2.1 Проектирование интерфейса пользователя 10

2.2 Разработка архитектуры программного обеспечения 11

2.3 Проектирование базы данных 13

3 Разработка и интеграция модулей программного обеспечения 14

3.1 Разработка программных модулей 14

3.2 Реализация интерфейса пользователя 16

3.3 Разграничение прав доступа пользователей 17

4 Тестирование и отладка программного обеспечения 18

4.1 Структурное тестирование 18

4.2 Функциональное тестирование 19

5 Инструкция по эксплуатации 23

5.1 Установка приложения 23

5.2 Инструкция по работе 24

Заключение 29

Список использованных источников 30

**ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ**

В настоящем курсовом проекте применяют следующие сокращения и обозначения:

БД – база данных

ОЗУ – оперативное запоминающее устройство

ОС – операционная система

ПО – программное обеспечение

СУБД – система управления базами данных

ЯП – язык программирования

API – интерфейс программирования приложения

ERD – диаграмма «сущность-связь»

IDE – интегрированная среда разработки

**ВВЕДЕНИЕ**

Актуальность разрабатываемого проекта обусловлена растущим интересом к рыбалке как к хобби и способу отдыха, а также необходимостью адаптации бизнеса к современным условиям рынка.

В последние годы наблюдается значительное увеличение числа людей, увлеченных рыбалкой, что делает эту деятельность не только популярным способом проведения свободного времени, но и важным элементом культурного досуга.

С увеличением числа рыболовов возрастает потребность в удобных и эффективных инструментах для оформления заказов на рыболовные товары. В условиях высокой конкуренции и разнообразия предложений важно создать систему, которая обеспечит пользователям простой и интуитивно понятный интерфейс для взаимодействия с поставщиками товаров. Это позволит не только упростить процесс покупки, но и повысить уровень удовлетворенности клиентов, что, в свою очередь, будет способствовать росту бизнеса.

Целью курсового проектирования является разработка многопользовательской клиент-серверной информационной подсистемы для учета заказов и управления ресурсами в сфере рыболовства. Эта система будет предназначена для обеспечения эффективного взаимодействия между рыболовами и администраторами, а также для автоматизации процессов оформления, обработки и учета заказов. Важно, чтобы система не только удовлетворяла текущие потребности пользователей, но и была готова к масштабированию и адаптации к будущим изменениям в рыночной среде.

Для достижения поставленной цели разработки многопользовательской клиент-серверной информационной системы для учета заказов и управления ресурсами в сфере рыболовства требуется решить следующие задачи:

* выполнить сбор требований целевой аудитории,
* проанализировать информационные источники по предметной области,
* спроектировать архитектуру приложения,
* спроектировать диаграмму вариантов использования приложения,
* выбрать состав программных и технических средств для реализации веб-приложения,
* спроектировать БД,
* создать БД в определенной СУБД,
* разработать API для веб-приложения,
* реализовать разграничение прав доступа пользователей,
* реализовать защиту данных,
* разработать пользовательский интерфейс веб-приложения,
* реализовать функционал оформления заказов,
* реализовать возможность просмотра и управления заказами,
* выполнить структурное тестирование ПО,
* выполнить функциональное тестирование ПО,
* разработать программную и эксплуатационную документацию.

В результате выполнения поставленных задач будет разработано веб-приложение для оформления заказов у рыболовов.

**1 Анализ и разработка требований**

**1.1 Назначение и область применения**

Разрабатываемое веб-приложение предназначено как для физических, так и для юридических лиц, желающих оптимизировать процесс оформления заказов на рыбу. В условиях растущего интереса к рыболовству и увеличения числа поставщиков, приложение станет незаменимым инструментом для пользователей, стремящихся упростить и ускорить процесс поиска и заказа рыбы.

**1.2 Постановка задачи**

Для разработки подсистемы оформления заказов для рыболовов необходимо создать функциональность, которая обеспечит удобный и интуитивно понятный интерфейс для пользователей. Подсистема предоставит доступ к следующей функциональности:

* авторизации,
* регистрации,
* создание и редактирование заказов,
* просмотр доступных товаров и спроса,
* фильтрация и поиск товаров,
* оформление заказа,
* отмена заказа.

**1.3 Описание алгоритма функционирования системы**

При запуске приложения отображается главная страница, на которой присутствует описание возможностей веб-приложения. Пользователь может зарегистрироваться или войти в систему. Гостям доступен только просмотр каталога товаров и запросов.

С главной страницы веб-приложения пользователи могут легко перейти на страницу каталога товаров, что обеспечивает удобный доступ к ассортименту. На странице каталога пользователи имеют возможность просматривать доступные товары, что позволяет им ознакомиться с предложениями и выбрать наиболее подходящие варианты.

Зарегистрированные пользователи имеют доступ к личному кабинету, который позволяет управлять своими товарами, запросами и заказами, а также просмотреть заказы оформленные на их товар. Они могут добавить или удалить свой товар, также могут оформить заказ, указав вес который хотят приобрести.

Администратор имеет доступ к панели управления, которая предоставляет ему широкий спектр возможностей для управления приложением. В этой панели администратор может удалять пользователей, что позволяет поддерживать актуальность и безопасность базы данных. Удаление неактивных или нарушающих правила пользователей помогает обеспечить порядок и защиту интересов всех участников платформы, а также добавить название рыбы.

На рисунке 1 изображена диаграмма вариантов использования мобильного приложения.

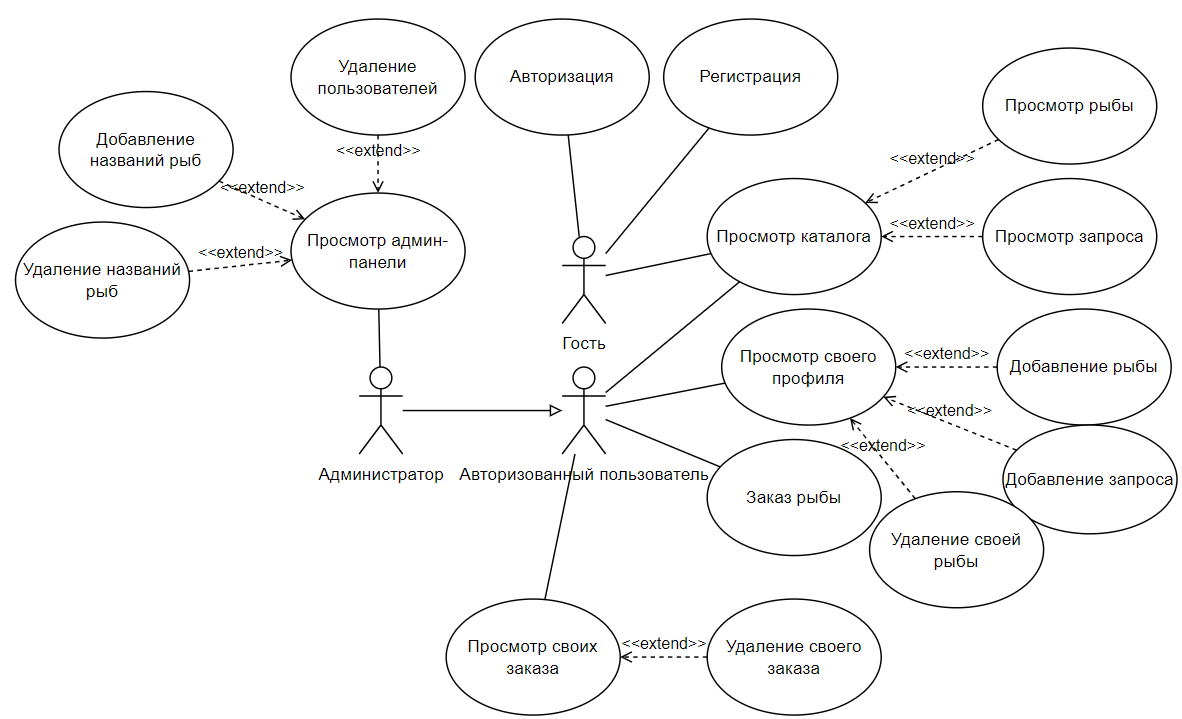


Рисунок 1 – Диаграмма вариантов использования веб-приложения

**1.4 Выбор состава программных и технических средств**

Согласно цели проекта, требуется создать веб-приложение для оформления заказов у рыболовов.

Работа с веб-приложением будет осуществляться на персональных компьютерах, которые соответствуют определенным системным требованиям. Пользователи смогут использовать устройства с установленной операционной системой Windows 10 и выше или macOS 10.14 и выше. Эти операционные системы обеспечивают стабильную и безопасную работу приложения, а также поддержку современных веб-технологий.

В качестве СУБД выбрана MySQL 8, так как она может использоваться как для небольших веб-приложений, так и для серьезных корпоративных систем, а также обладает хорошей производительностью и быстродействием благодаря оптимизированным алгоритмам выполнения запросов[1].

Приложение будет написано на ЯП JavaScript с использованием фреймворка Next.js, так как у него есть активное сообщество и множество доступных плагинов и библиотек, что облегчает интеграцию с другими инструментами и технологиями, а также включает в себя множество встроенных функций для оптимизации производительности, таких как автоматическая разбивка кода, оптимизация изображений и предварительная загрузка страниц.

Для разработки приложения будет использоваться IDE Visual Studio Code версии 1.60 и выше, так как он имеет обширный каталог расширений, которые позволяют добавлять новые функции, такие как поддержка фреймворков, инструментов для тестирования, интеграции с системами контроля версий, имеет встроенную поддержку Git, что упрощает управление версиями кода и много другого.

Для функционирования системы на стороне сервера достаточны следующие программные и технические средства:

* ОС Ubuntu версии 20.04 или выше,
* сервер БД: MySQL версии 8.0 и выше,
* процессор с частотой 2 ГГц,
* свободная оперативная память объемом 1 ГБ,
* ПО для конфигурирования, управления и администрирования сервера БД: MySQL Workbench версии 8.0 и выше,
* ПО для работы API: PHP версии 8.0 и выше, фреймворк Laravel версии 9.0 и выше для создания RESTful API,

Для функционирования системы на стороне клиента достаточны следующие программные и технические средства:

* ОС Windows 10 (версии 64-бит) или macOS 10.14 и выше,
* веб-браузер: Google Chrome (версии 90 и выше) или Mozilla Firefox (версии 88 и выше) для корректного отображения веб-приложения,
* процессор с частотой 2 ГГц,
* оперативная память: Минимум 4 ГБ,
* свободное место в хранилище: 200 МБ,
* постоянное интернет-подключение.

Эти программные и технические средства обеспечат эффективное функционирование веб-приложения для оформления заказов у рыболовов.

**2 Проектирование программного обеспечения**

**2.1 Проектирование интерфейса пользователя**

В рамках разработки веб-приложения был создан интерфейс пользователя в виде wireframe с использованием платформы Figma. Этот инструмент позволяет разработчикам и дизайнерам визуализировать структуру приложения, что является важным этапом в процессе проектирования. Wireframe представляет собой схематическое изображение, которое демонстрирует основные элементы интерфейса, такие как кнопки, меню, формы и другие компоненты, а также их расположение на экране.

Wireframe каталога веб-приложения представлен на рисунке 2.

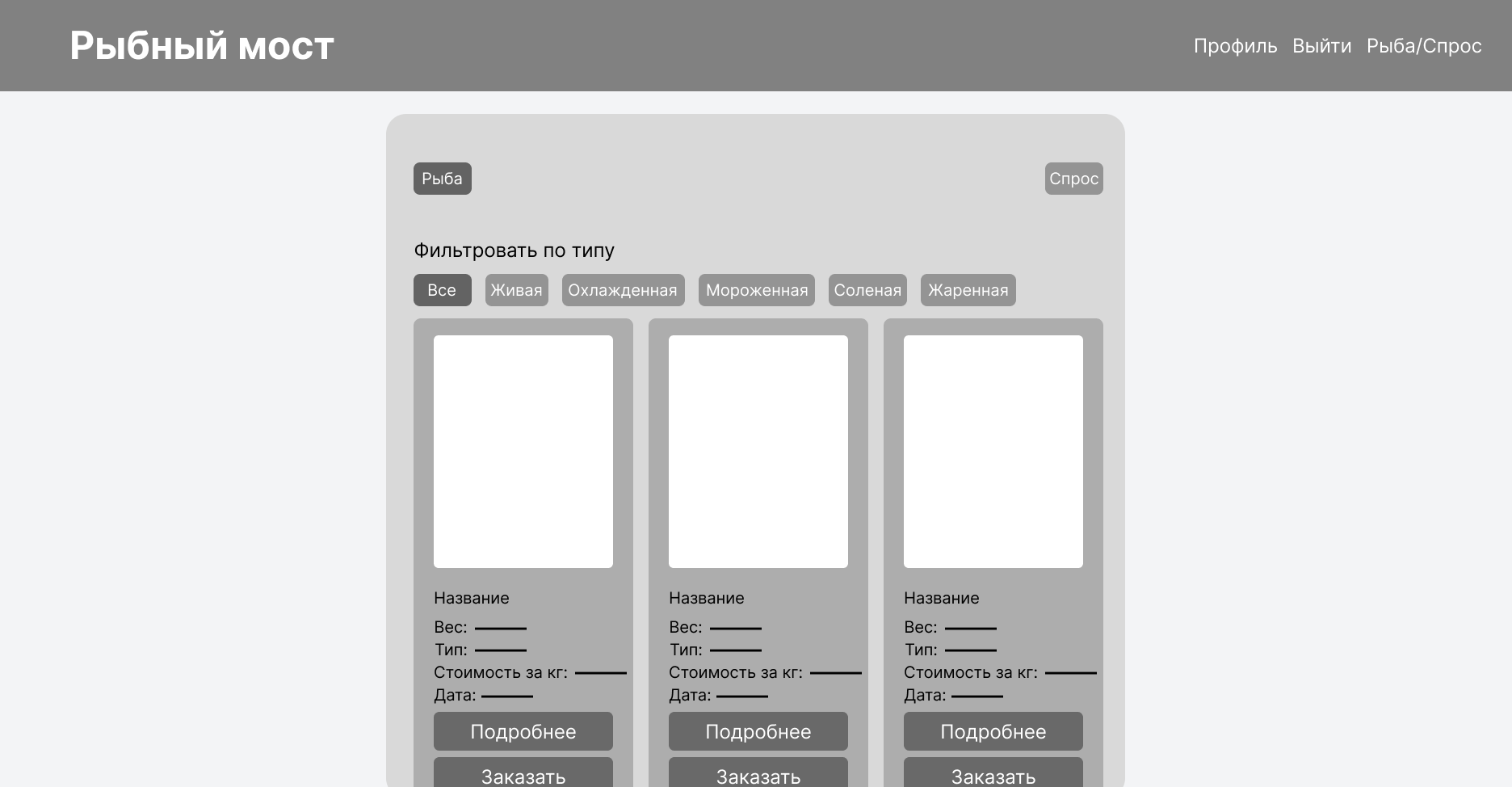


Рисунок 2 – Figma. Wireframe интерфейса каталога

**2.2 Разработка архитектуры программного обеспечения**

Приложение для оформления заказов у рыболовов предназначено для соединения пользователей с товарами и услугами, связанными с рыбалкой. Архитектура приложения построена на основе клиент-серверной модели и включает в себя несколько ключевых компонентов: серверная часть приложения, клиентская часть приложение, БД[3].

Для разрабатываемой подсистемы разработана диаграмма развертывания, представленная на рисунке 3.

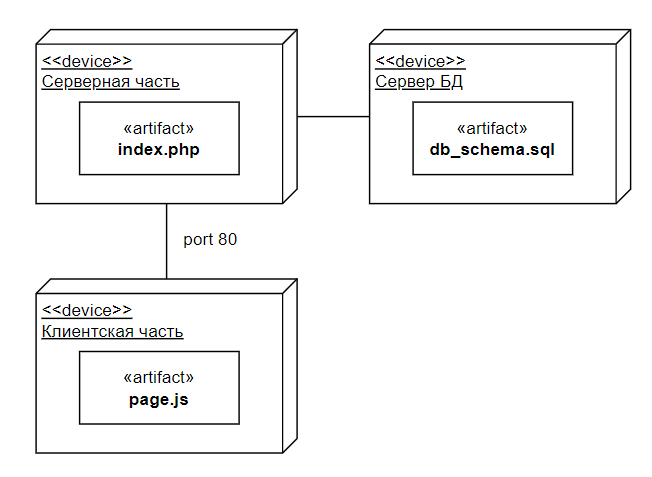


Рисунок 3 – draw.io. Диаграмма развертывания

**2.3 Проектирование базы данных**

В рамках курсового проектирования требуется разработать БД для подсистемы оформления заказов у рыболовов. которая будет использоваться как физическими, так и юридическими лицами, необходимо определить основные сущности и их взаимосвязи[1].

Модели БД созданы при помощи draw.io. На рисунке 4 в виде ERD показана часть физической модели предметной области.

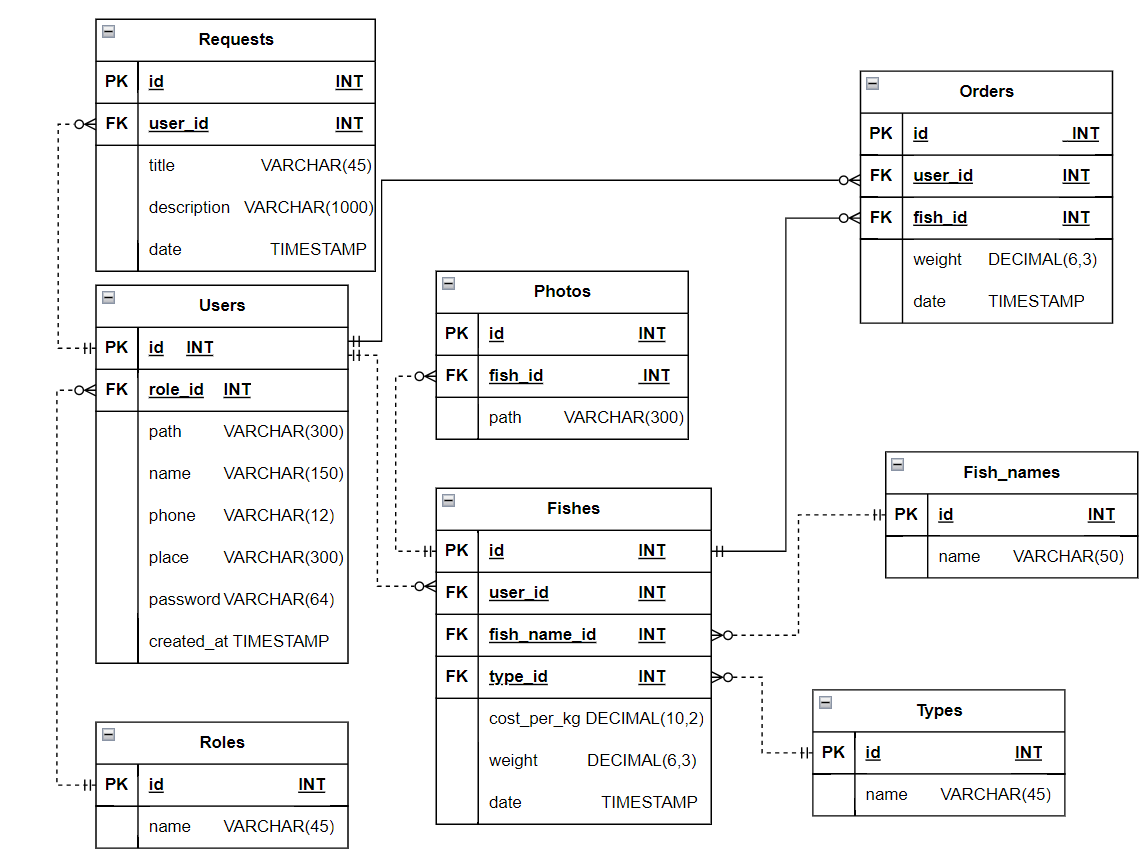


Рисунок 4 – draw.io. Физическая модель БД

**3 Разработка и интеграция модулей программного обеспечения**

**3.1 Разработка программных модулей**

В ходе курсового проектирования были разработаны: веб-приложение на JavaScript с использованием фреймворка Next JS и API приложение на PHP с использованием фреймворка Laravel в IDE Visual Studio Code.

Взаимодействие мобильного приложения с сервером будет происходить при помощи HTTP-запросов к API, ответы будут возвращатся в формате JSON. Для реализации HTTP-запросов использован сетевой клиент http. Код метода для получения списка рыб отправкой GET-запроса на сервер представлен листингом 1[5].

Листинг 1 – Код метода для отправки GET-запроса на сервер

|  |
| --- |
| // Функция для получения списка рыб с сервера  export const getFishes = async () => {      try {          // Выполняем GET-запрос к эндпоинту '/fishes'          // с помощью axios          const response = await axios.get('/fishes');            // Возвращаем данные о рыбах из ответа сервера          return response.data.data;      } catch (error) {          // Обрабатываем ошибку, если запрос не удался          handleError(error);      } |
| }; |

Для аутентификации на стороне сервера была разработана функция представленная листингом 2.

Получение информации из БД осуществляется посредством API приложения, код метода получения списка рыбы относящейся к определенному пользователю представлен листингом 3.

Листинг 2 – Код функции аутентификации

|  |
| --- |
| public function store(Request $request)      {          try {              // Валидация входящих данных              $request->validate([                  'phone' => 'required|string',                  'password' => 'required|string',              ]);              // Извлечение учетных данных из запроса              $credentials = $request->only('phone', 'password');              \Log::info('Попытка аунтефикации для телефона: ', ['phone' => substr($credentials['phone'], -4)]);              // Попытка аутентификации пользователя с использованием предоставленных учетных данных              if (!Auth::attempt($credentials)) {                  return response()->json(['error' => 'invalid\_credentials'], 401);              }              // Если аутентификация успешна, создаем ресурс //пользователя              $user = new UserResources(Auth::user());              return response()->json(['user' => $user]);          } catch (\Exception $e) {              // Логируем ошибку аутентификации              \Log::error('Ошибка аунтефикации: ' . $e->getMessage());              return response()->json(['error' => 'internal\_server\_error'], 500);          } |
| } |

Листинг 3 – Код метода получения списка рыбы относящейся к определенному пользователю

|  |
| --- |
| // Метод для получения всех рыб, принадлежащих пользователю с // заданным идентификатором  public function fishes($id)  {  // Выполняем запрос к модели Fish, чтобы получить все  // записи, где user\_id соответствует переданному  // идентификатору  // Возвращаем коллекцию ресурсов FishResources, которая  // форматирует данные для ответа      return FishResources::collection(Fish::where('user\_id', '=', $id)->get()); |
| } |

**3.2 Реализация интерфейса пользователя**

Для реализации интерфейса пользователя были использованы следующие технологии:

* HTML/CSS: для разметки и стилизации страниц.
* JS: для динамического взаимодействия с пользователем.
* Next.js: для создания компонентов интерфейса и управления состоянием приложения.

Для отображения информации о заказе был разработан компонент RequestCard, который представлен на рисунке 5, его код представлен листингом 3.

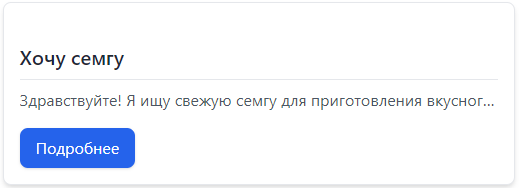


Рисунок 5 – Рыбный мост. Компонент FishCard

Листинг 3 – Код компонента элемента списка запросов

|  |
| --- |
| import Link from 'next/link';  const RequestCard = ({ request }) => {  return (          <div key={request.id}              className="border rounded-lg p-4 shadow-md transition-transform transform hover:scale-105 bg-white hover:bg-gray-100 mb-4 max-w-sm mx-auto sm:max-w-md md:max-w-lg lg:max-w-xl"              style={{ width: '576px', height: '183px' }} // Устанавливаем стандартные размеры          > |
| <div className="mt-6">                  <h2 className="text-lg md:text-xl font-semibold text-gray-800">{request.title}</h2>                  <div className="border-t border-gray-200 mt-2 pt-2">                      <div className="text-gray-600 overflow-hidden text-ellipsis whitespace-nowrap max-w-full"                      >                          {request.description}                      </div>                  </div>              </div>              <div className="mt-4">                  <Link href={`/request/${request.id}`}  className="inline-block bg-blue-600 text-white py-2 px-4 rounded-lg shadow hover:bg-blue-700 transition duration-200 text-center">                      Подробнее                  </Link>              </div>          </div>      );  };  export default RequestCard; |

**3.3 Разграничение прав доступа пользователей**

Разграничение прав доступа реализовано при помощи таблицы roles. Права администратора у пользователя появляются при role\_id равном 2.

Пример кода разграничения прав пользователей веб-приложения представлен листингом 4.

Листинг 4 – Код разграничения прав пользователей

|  |
| --- |
| const isAdmin = () => {  return user && user.role\_id !== 1; // Если роль не 1,  // значит это администратор |
| }; |

**4 Тестирование и отладка программного обеспечения**

**4.1 Структурное тестирование**

Во время курсового проектирования проведено тестирование методом белого ящика для страницы профиля, результаты в таблице 1.

Таблица 1 – Тестирование страницы профиля

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Фактический результат |
| Войти на страницу без авторизации | Перенаправление на авторизацию | Совпадает с ожидаемым |
| Войти на страницу с авторизацией | Появление сообщения об успешной авторизации | Совпадает с ожидаемым |
| Нажать на кнопку «Мои заказы» | Перенаправление на страницу с заказами пользователя | Совпадает с ожидаемым |
| Нажать на кнопку «Заказы» | Перенаправление на страницу с заказами пользователей | Совпадает с ожидаемым |
| Нажать на кнопку «Спрос» | Отображение запросов пользователя | Совпадает с ожидаемым |
| Нажать на кнопку «Рыбы» | Отображение рыбы пользователя | Совпадает с ожидаемым |
| Нажать на кнопку «Добавить запрос» | Перенаправление на страницу создания запроса | Совпадает с ожидаемым |
| Нажать на кнопку «Добавить рыбу» | Перенаправление на страницу создания рыбы | Совпадает с ожидаемым |
| Нажать на кнопку «Выйти» | Перенаправление на страницу авторизации | Совпадает с ожидаемым |

**4.2 Функциональное тестирование**

Во время курсового проектирования проведено функциональное тестирование страницы авторизации, каталога, добавления рыбы и запроса методом черного ящика, результаты тестирования в таблицах 2-5.

Таблица 2 – Тестирование страницы авторизации методом черного ящика

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Фактический результат |
| Нажать на кнопку «Войти» без введеных данных | Появление ошибки о том что необходимо заполнить поля | Совпадает с ожидаемым |
| Нажать на кнопку «Войти», ввести «+79502537238» в поле ввода номер телефона, ввести «1234qwer» в поле ввода пароля | Перенаправление на страницу профиля | Совпадает с ожидаемым |
| Нажать на кнопку «Войти», ввести «+79502537238» в поле ввода номер телефона, ввести «1234» в поле ввода пароля | Появление ошибки о том что пароль неверный | Совпадает с ожидаемым |
| Нажать на кнопку «Войти», ввести «89502537238» в поле ввода номер телефона, ввести «1234» в поле ввода пароля | Появление ошибки о том что логин неверный | Совпадает с ожидаемым |
| Нажать на кнопку «Зарегестрироваться» | Переход на страницу регистрации | Совпадает с ожидаемым |
| Нажать на надпись «Рыбный мост» | Переход на главную страницу | Совпадает с ожидаемым |
| Нажать на кнопку «Рыба/Спрос» | Переход на страницу каталога | Совпадает с ожидаемым |

Таблица 3 – Тестирование страницы каталога методом черного ящика

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Фактический результат |
| Нажать на кнопку «Запросы» | Отображение запросов | Совпадает с ожидаемым |
| Нажать на кнопку «Подробнее» | Перенаправление на страницу рыбы | Совпадает с ожидаемым |
| Нажать на кнопку «Заказать» без авторизации | Перенаправление на страницу авторизации | Совпадает с ожидаемым |
| Нажать на кнопку «Заказать» с авторизацией | Перенаправление на страницу оформления заказа | Совпадает с ожидаемым |
| Нажать на кнопку «Живая» | Отображение рыбы с типом «Живая» | Совпадает с ожидаемым |
| Нажать на кнопку «Охлажденная» | Отображение рыбы с типом «Охлажденная» | Совпадает с ожидаемым |
| Нажать на кнопку «Мороженная» | Отображение рыбы с типом «Мороженная» | Совпадает с ожидаемым |
| Нажать на кнопку «Соленая» | Отображение рыбы с типом «Соленая» | Совпадает с ожидаемым |
| Нажать на кнопку «Сушеная» | Отображение рыбы с типом «Сушеная» | Совпадает с ожидаемым |
| Нажать на кнопку «Жаренная» | Отображение рыбы с типом «Жаренная» | Совпадает с ожидаемым |
| Нажать на кнопку «Запросы», нажать на кнопку «Подробнее» | Перенаправление на страницу запроса | Совпадает с ожидаемым |

Таблица 4 – Тестирование страницы добавления рыбы методом черного ящика

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Фактический результат |
| Нажать на кнопку «Добавить» без данных | Появление ошибки о том что необходимо заполнить поля | Совпадает с ожидаемым |
| Выбрать тип «Живая» и название рыбы «Вобла» и ввести вес 123, цену за килограмм 60, нажать на кнопку «Добавить» | Перенаправление на страницу профиля и добавление рыбы без фотографий | Совпадает с ожидаемым |
| Выбрать тип «Живая» и название рыбы «Вобла» и ввести вес 123, цену за килограмм 60, нажать на кнопку «Добавить» | Перенаправление на страницу профиля и добавление рыбы с фотографиями | Совпадает с ожидаемым |
| Нажать на кнопку «Назад» | Перенаправление на страницу профиля | Совпадает с ожидаемым |
| Выбрать тип «Живая» и название рыбы «Вобла» и ввести вес 123, нажать на кнопку «Добавить» | Появление ошибки о том что необходимо заполнить поля | Совпадает с ожидаемым |

Таблица 5 – Тестирование страницы добавления запроса методом черного ящика

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Фактический результат |
| Нажать на кнопку «Добавить» без данных | Появление ошибки о том что необходимо заполнить поля | Совпадает с ожидаемым |
| Ввести название «Хочу семги», нажать на кнопку «Добавить» | Появление ошибки о том что необходимо заполнить поля | Совпадает с ожидаемым |

Продолжение таблицы 5

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Фактический результат |
| Ввести описание «Хочу семги», нажать на кнопку «Добавить» | Появление ошибки о том что необходимо заполнить поля | Совпадает с ожидаемым |
| Нажать на кнопку «Назад» | Перенаправление на страницу профиля | Совпадает с ожидаемым |

**5 Руководство пользователя**

**5.1 Установка приложения**

Для функционирования системы на стороне сервера достаточны следующие программные и технические средства:

* ОС: Ubuntu 20.04 или Windows 10,
* ОС: Linux (Ubuntu 20.04 или выше, Debian 10 или выше)
* процессор: 2 ГГц
* оперативная память: 4 ГБ
* свободное место на диске: 20 ГБ
* дополнительные компоненты: PHP 8.0 (или выше), MySQL 8.0 (или выше), Laravel 8.x (или выше), Nginx (или Apache), Composer для управления зависимостями PHP.

Добавление объектов БД происходит через миграции Laravel.

Для функционирования веб-приложения достаточны следующие программные и технические средства:

* ОС: Windows 10, macOS или Linux,
* ЯП: Node.js 14.x или выше,
* процессор: 2 ГГц,
* оперативная память: 4 ГБ,
* свободное место в хранилище: 500 МБ,
* оперативная память: 4 GB ОЗУ,
* дополнительно: npm для управления зависимостями JavaScript.

В веб-приложении используются следующие данные для авторизации:

* номер телефона – +79502537238,
* пароль – 1234qwer.

**5.2 Инструкция по работе**

Веб-приложение предназначено для взаимодействия между рыболовами и пользователями, желающими приобрести рыбу. Оно включает в себя функционал для регистрации, авторизации, оформления заказов, а также управления записями о рыбе.

После открытия веб-приложения пользователь попадает на главную страницу, где представлена информация о доступных функциях и последних записях о рыбе. На главной странице также находятся ссылки на страницы регистрации и авторизации. Главная страница представлена на рисунке 6.

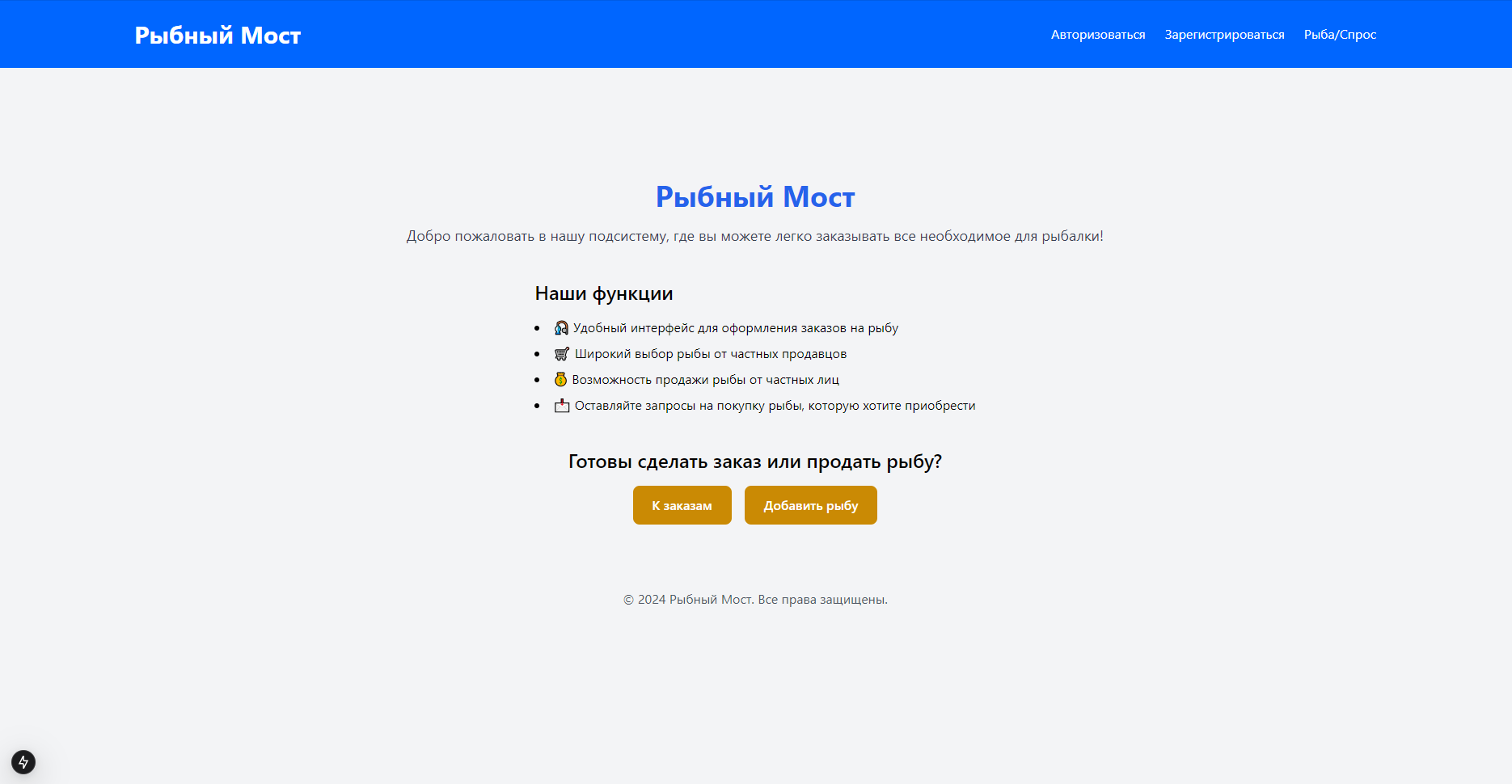


Рисунок 6 – Рыбный мост. Вид главной страницы

Пользователь может перейти на страницу регистрации или авторизации, она представлена на рисунке 7, где ему необходимо заполнить форму с личными данными, такими как имя, номер телефона и пароль. После успешной регистрации пользователь получает доступ к функционалу приложения.

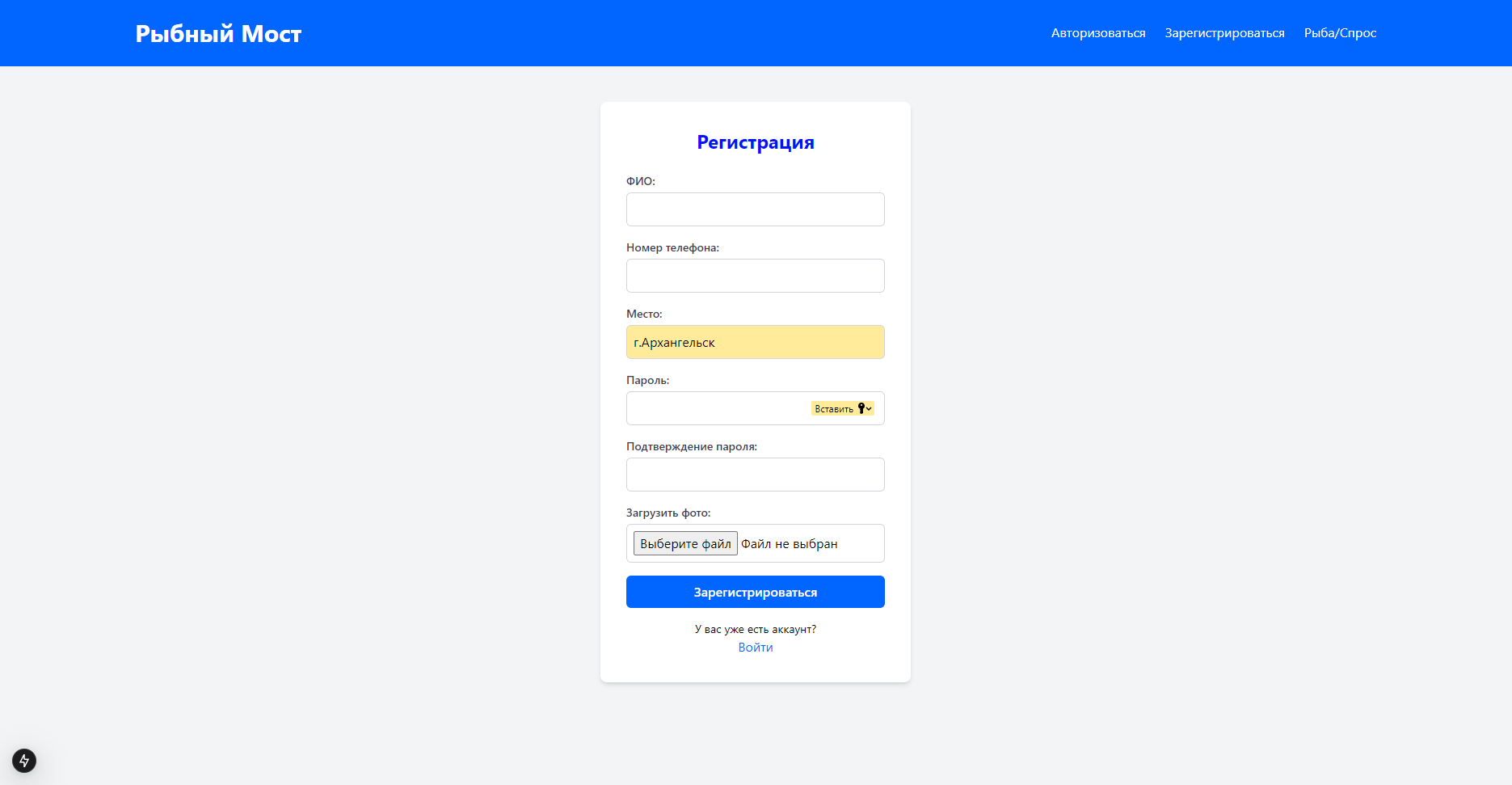


Рисунок 7 – Рыбный мост. Вид страницы регистрации

После авторизации или регистрации, пользователь, попадает на страницу профиля, представленную на странице 8.

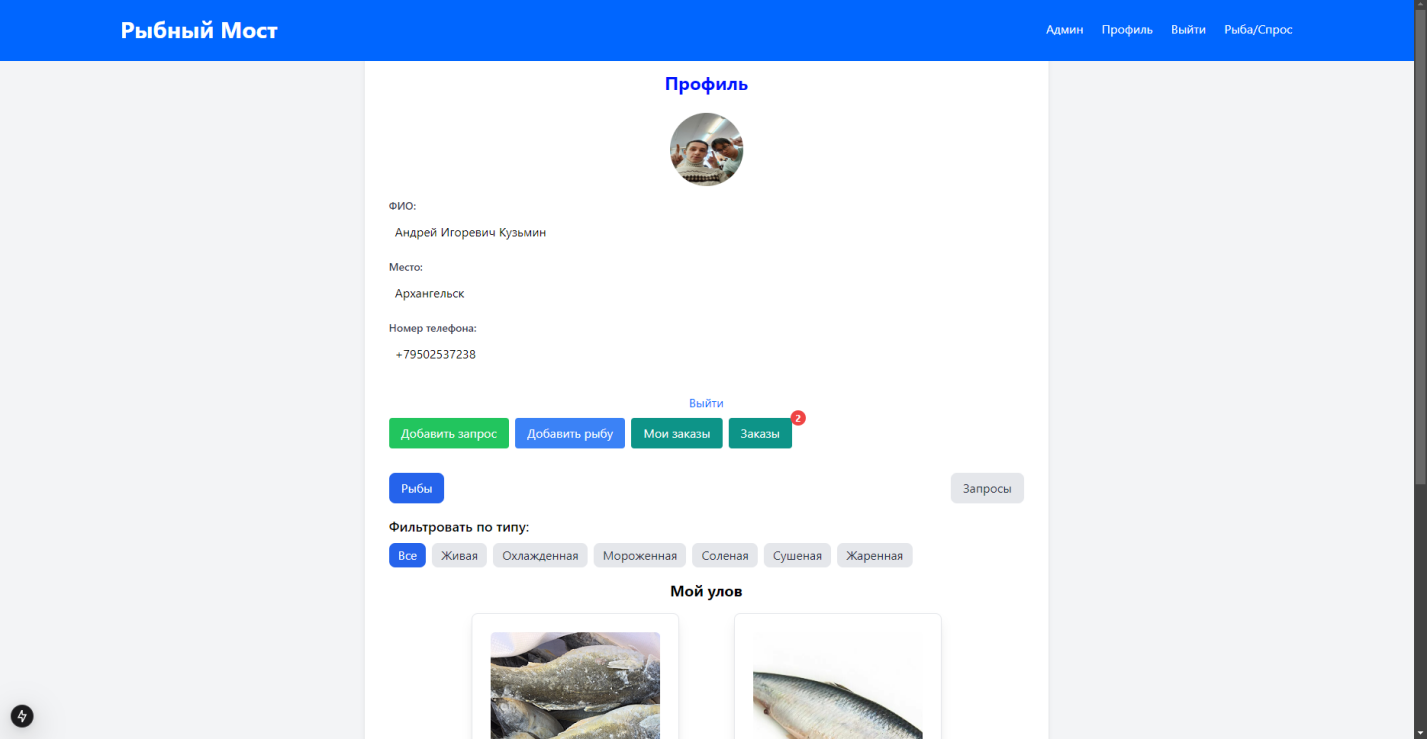


Рисунок 8 – Рыбный мост. Вид страницы профиля

Со страницы каталога, представленную на странице 9, пользователь, может просмотреть доступные записи о рыбе, оставленные другими пользователями. Он может выбрать рыбу и оформить заказ, заполнив форму с указанием веса. Страница оформления заказа представлена на рисунке 10.

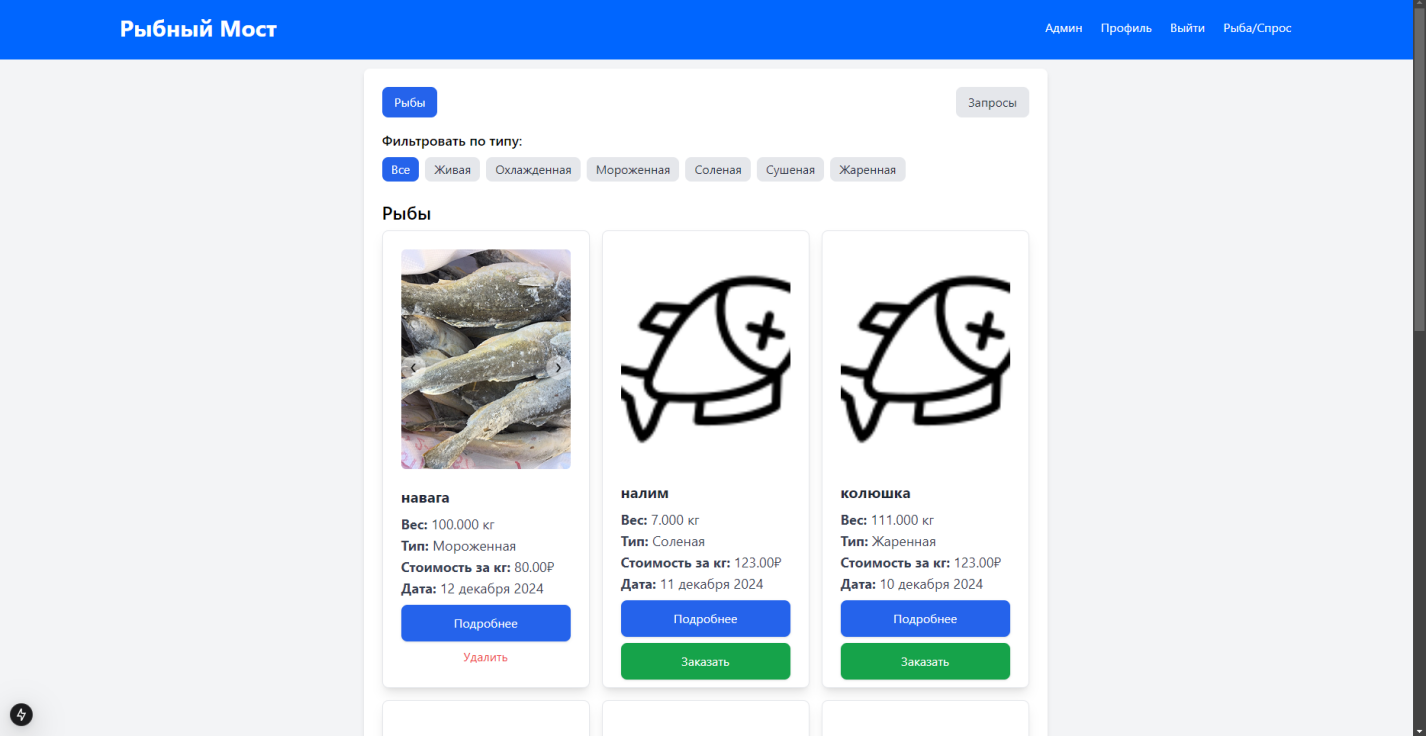


Рисунок 9 – Рыбный мост. Вид страницы каталога

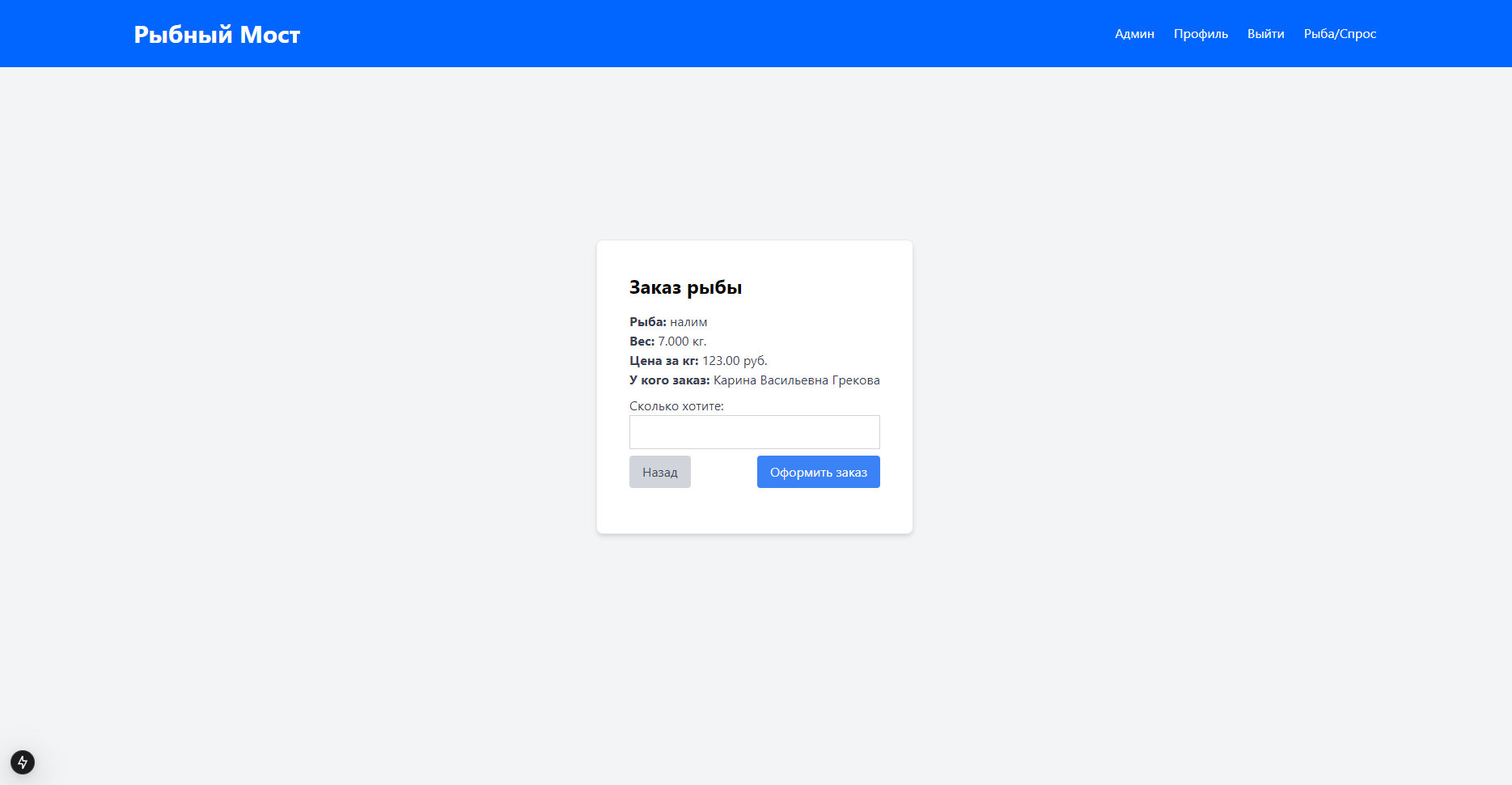


Рисунок 10 – Рыбный мост. Вид страницы оформления заказа

Если пользователь не нашел нужную рыбу, он может оставить запрос, написав в описании желаемую информацию о рыбе. Запрос будет виден другим пользователям, которые могут откликнуться на него. Страница создания запроса представлена на рисунке 11.

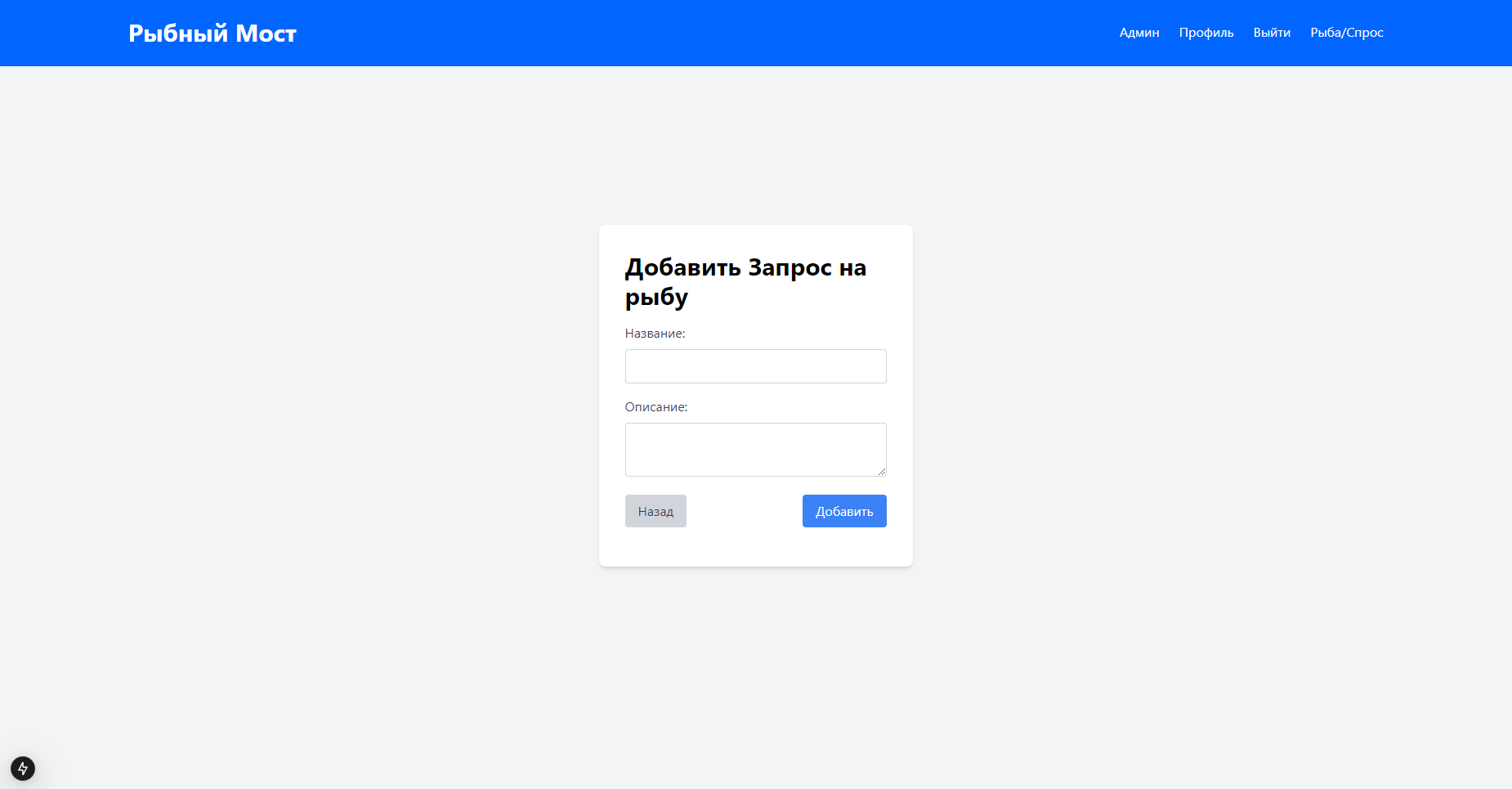


Рисунок 11 – Рыбный мост. Вид страницы создания запроса

Если пользователь является рыболовом, он может оставить запись о пойманной рыбе, добавив описание и фотографии. Это позволяет другим пользователям видеть доступные предложения. Форма создания записи о пойманной рыбе представлена на рисунке 12.

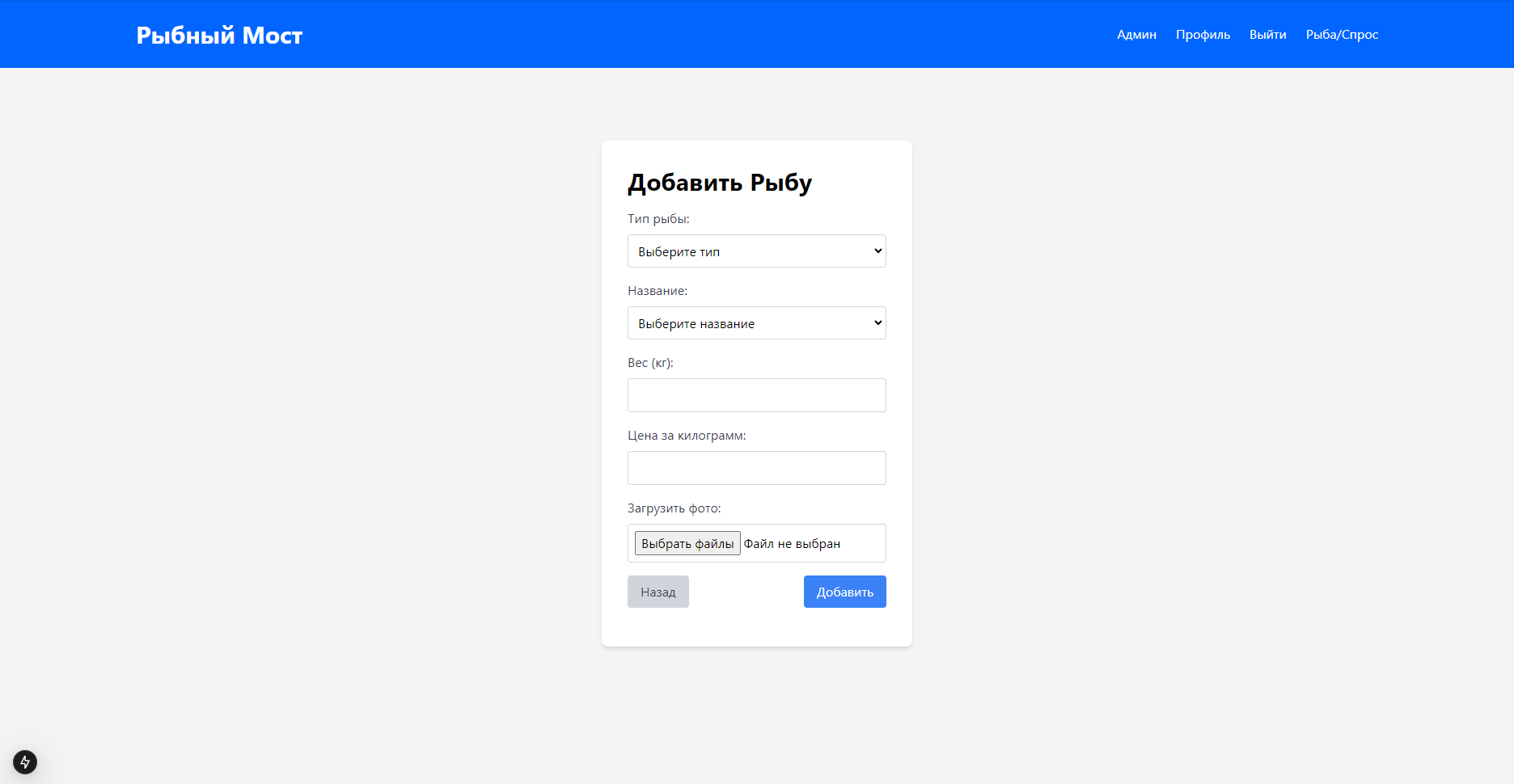


Рисунок 12 – Рыбный мост. Вид страницы создания записи о рыбе

При загрузке фотографий система проверяет формат и размер файлов, уведомляя пользователя о недопустимых значениях.

Администратор имеет доступ к полному функционалу управления приложением. Он может просматривать список заказов, рыб, запросов и пользователей. Также администратор может удалять записи, которые были сделаны давно или нарушают правила приложения.

Веб-приложение для рыболовов предоставляет пользователям удобный интерфейс для оформления заказов, оставления запросов и записей о рыбе. Администраторы имеют возможность управлять контентом и пользователями, обеспечивая порядок и актуальность информации. Система включает в себя механизмы обработки ошибок, что делает взаимодействие с приложением более удобным и безопасным.

**ЗАКЛЮЧЕНИЕ**

В ходе курсового проектирования достигнута поставленная цель: спроектировано веб-приложение для оформления заказов у рыболовов. Данное веб-приложение поможет пользователям эффективно взаимодействовать с поставщиками рыболовных товаров. Кроме того, решены все поставленные задачи:

* выполнен сбор требований целевой аудитории,
* проанализированы информационные источники по предметной области,
* спроектирована архитектура приложения,
* спроектирована диаграмма вариантов использования приложения,
* выбран состав программных и технических средств для реализации веб-приложения,
* спроектирована БД,
* создана БД в определенной СУБД,
* разработано API для веб-приложения,
* реализовано разграничение прав доступа пользователей,
* реализовано защиту данных,
* разработан пользовательский интерфейс веб-приложения,
* реализована функциональность оформления заказов,
* реализована возможность просмотра и управления заказами,
* выполнено структурное тестирование ПО,
* выполнено функциональное тестирование ПО,
* разработана программная и эксплуатационная документация.

В результате выполнения поставленных задач разработано веб-приложение для оформления заказов у рыболовов, отвечающее современным тенденциям и требованиям заказчика.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Голицына, О. Л. Базы данных : учебное пособие / О. Л. Голицына, Н. В. Максимов, И. И. Попов. – 4-е изд., перераб. и доп. – Москва : ФОРУМ : ИНФРА-М, 2020. – 400 с. – URL: https://znanium.com/catalog/document?id=362825. – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.
2. Голицына, О. Л. Основы проектирования баз данных : учебное пособие / О. Л. Голицына, Т. Л. Партыка, И. И. Попов. – 2-е изд., перераб. и доп. Москва : ФОРУМ : ИНФРА-М, 2021. – 416 с. – URL: https://znanium.com/catalog/document?id=364900. – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.
3. Мартишин, С. А. Базы данных. Практическое применение СУБД SQL и NoSOL-типа для применения проектирования информационных систем : Учебное пособие / С. А. Мартишин, В. Л. Симонов, М. В. Храпченко. – Москва : ИД «ФОРУМ» : ИНФРА-М, 2018. – 368 с. – URL: http://znanium.com/bookread2.php?book=926871. – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.
4. Минимальная ширина – Текст : электронный // https://tailwindcss.ru : [сайт]. – 2024. – URL: https://tailwindcss.ru/docs/min-width/ (дата обращения: 27.11.2024).
5. Выборка и кэширование данных – Текст : электронный // https://nextjs.org : [сайт]. – 2024. – URL: https://nextjs.org/docs/app/building-your-application/data-fetching/fetching (дата обращения: 30.11.2024).