**РУКОВОДСТВО ПРОГРАММИСТА**

**Веб-приложение**

**«Find your pet»**

Киров

2022

**АННОТАЦИЯ**

В данном программном документе приведено руководство программиста по работе с веб-приложением «Find your pet».

В разделе «Назначение и условия применения программы» указаны назначение и функции, выполняемые сайтом, условия, необходимые для его работы.

В разделе «Характеристика приложения» приведено описание основных

характеристик и особенностей.

1. Назначение и условия применения программы

Домашние питомцы — наши лучшие друзья. Забота о них доставляет радость и удовольствие. На протяжении всего существования человечества, люди заводят различных питомцев и зачастую они по тем или иным причинам теряются или пропадают.

Главной задачей, которую необходимо было выполнить, было создание веб-приложения «Find your pet».

Продукт предназначен для помощи людям, попавшим в трудную ситуацию с пропажей домашнего любимца.

В ходе учебной практики была создано веб-приложение «Find your pet», предназначенная для помощи в размещении заявок о пропавших питомцах.

1. Характеристики и работа программы

Требования к приложению, которые были выполнены в ходе разработки:

1. Запуск возможен с любой «Windows» от 2007 года выпуска.
2. Наличие браузера на ПК

Требования к логике программы, которые были выполнены в ходе разработки:

1. Вход на сайт легко доступен для любого пользователя
2. Авторизация или создание новой учётной записи не обязателен для размещения заявки
3. Интуитивное главное меню со всеми необходимыми пунктами
4. При отправке заявки показывает ее в сайдбаре
5. Возможность смотреть заявки других пользователей и их профили

Для того, чтобы собрать приложение требуется:

1. Скачать и установить Git, по следующей ссылке

https://git-scm.com/download/win/

1. После установки Git, нужно открыть командную строку, создать папку, где будет находиться проект, далее нужно написать git clone https://github.com/RIXRIP/animals.git и дождаться завершения загрузки проекта.
2. Установить любую IDE, поддерживающую JS , Node.js, рекомендуется MS Visual Studio Code
3. Открыть папку через установленную IDE и открыть папку с проектом

На рисунке 1 представлена файловая архитектура приложения.

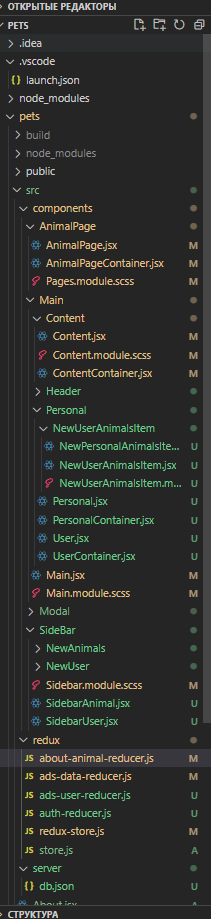


Рисунок 1 – Файловая архитектура приложения

Для работы необходимы все файлы, представленные на рисунке 1, их можно разделить на следующие группы:

1. Компоненты
2. Ресурсы
3. Стили
4. json файлы
5. Редьюсеры

При создании веб-приложения использовался язык JS и фреймворк ReactJS в связке с Redax, а так же модуль scss

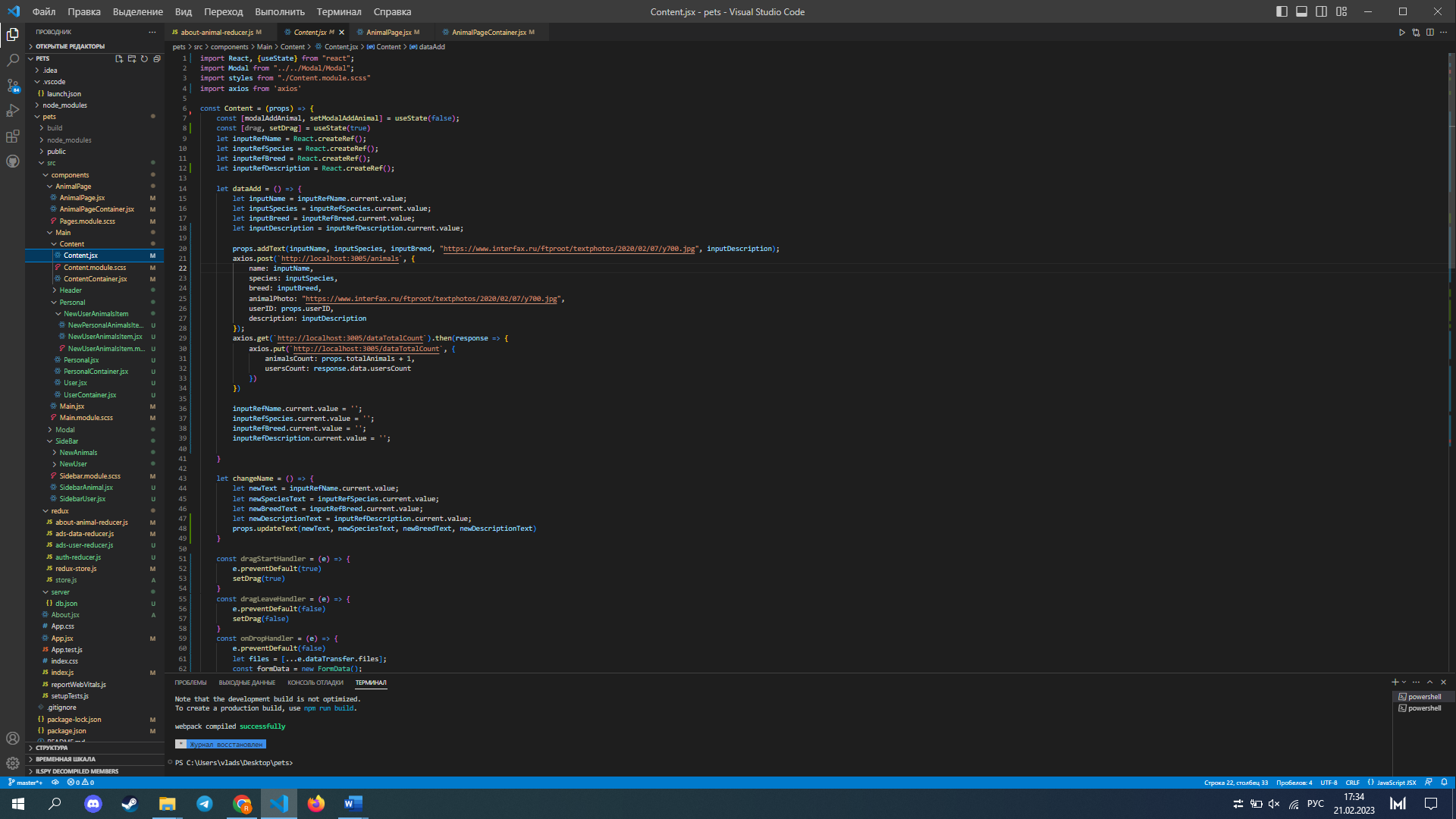
После запуска сайта становится доступен весь функционал приложения. На рисунке 2 представлен код презентационной компоненты Content. 

Рисунок 2 – Код презентационной компоненты Content.

При создании использовался фреймворк ReactJS с Redux. Основная компонента где происходит отрисовка – Content. В ней описаны get и post запросы к локальному json-server, выполняемые Axios. А так же тут описана основная разметка приложения.

При создании приложения была реализована модальная форма для последующего ее переиспользоваия:

На рисунке 2 главная старница.

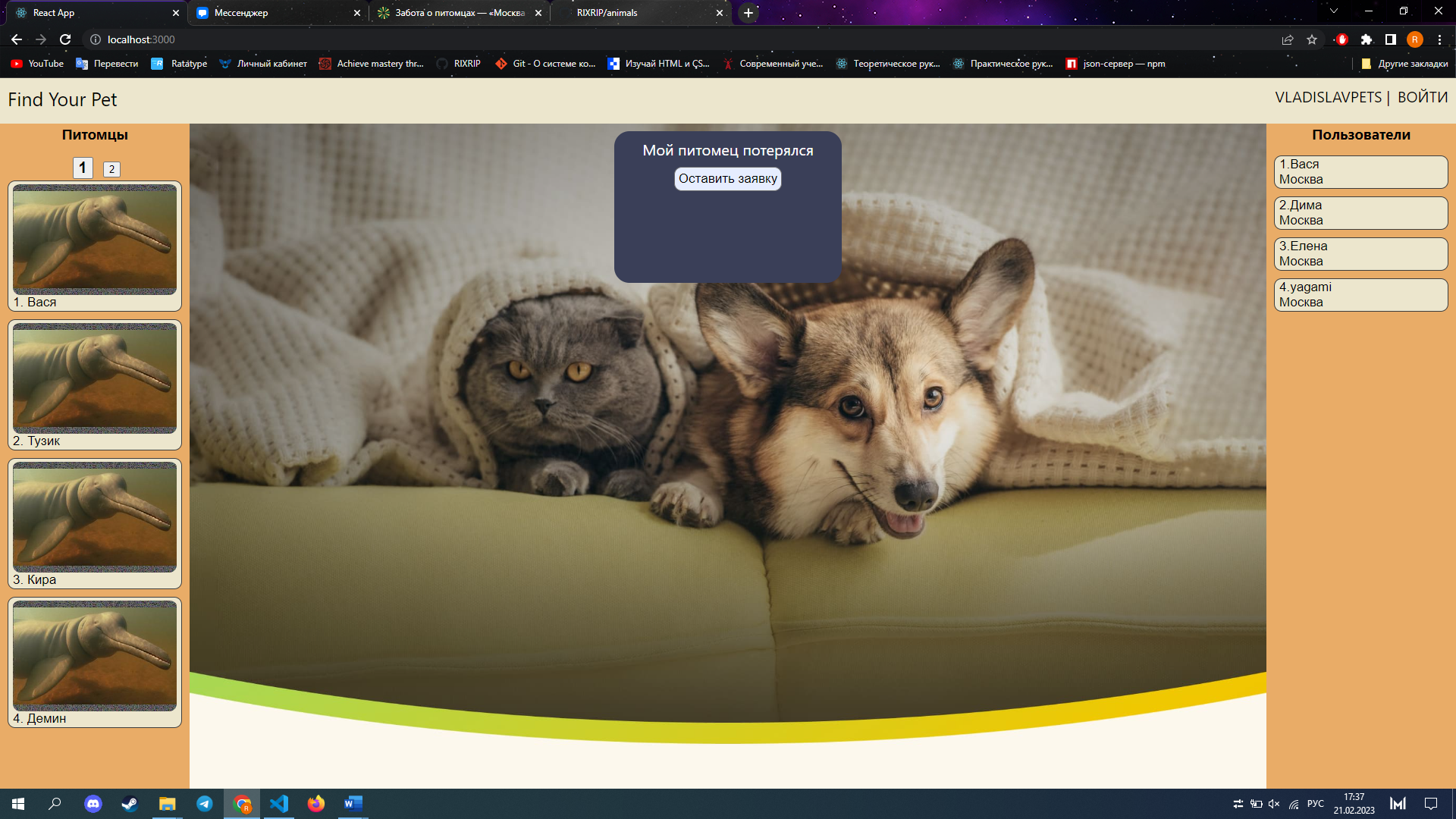


Рисунок 3 – Главная старница

Проверка нажатий на различные формы и кнопки осуществлена с помошью метода onClick с описаной внутри него логикой выполнения.

На рисунке 4 предсталена реализация модального окна.

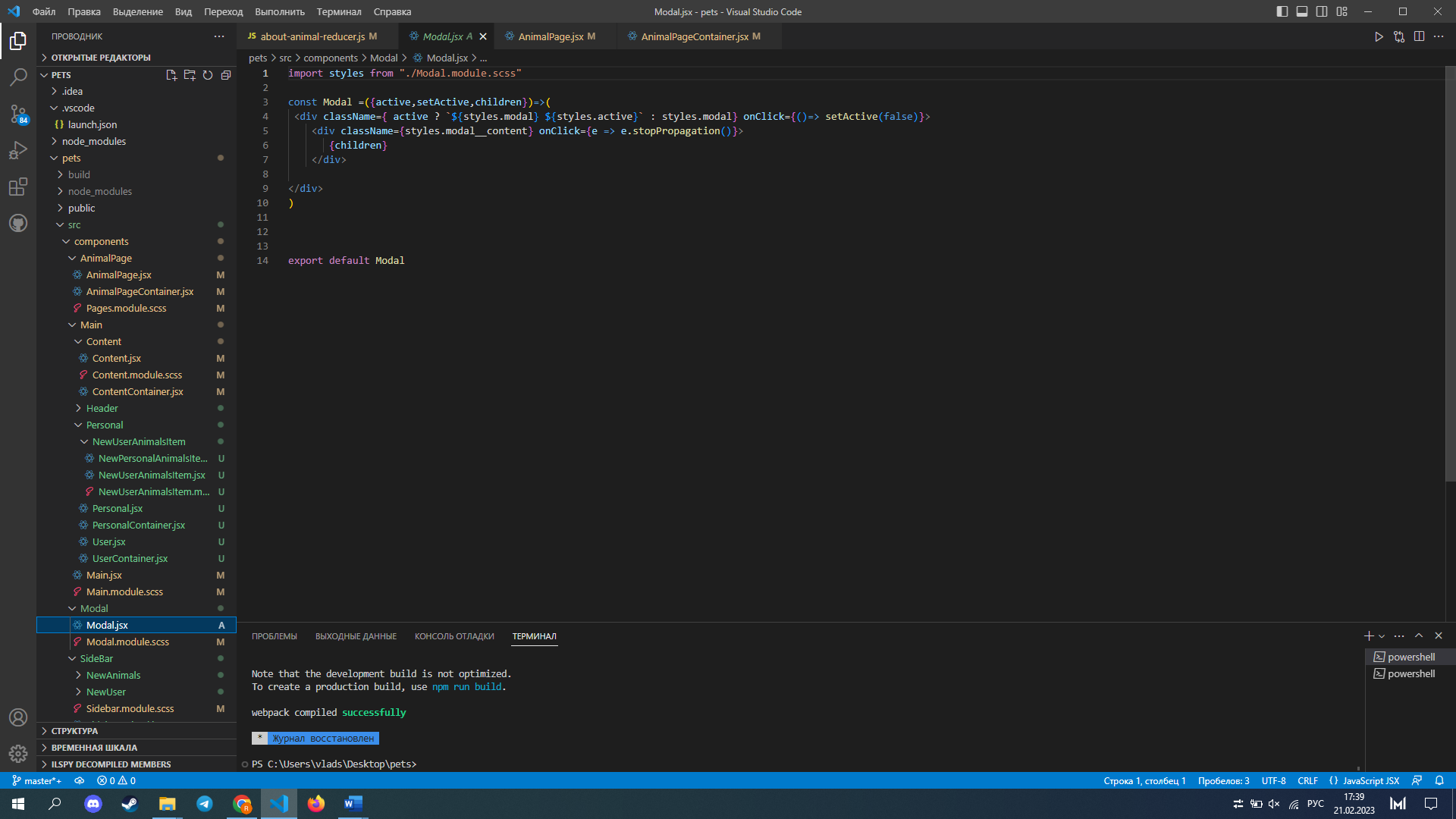


Рисунок 4 – Реализация модального окна

На рисунке 5 представлена реализация reducer. В нем реализован initialState с объектами, а так же сам редьюсер принимающий state и action. Внутри Switch реализованы case методы для записи данных в state. showAnimal является action creater, в него приходят данные из компонент, которые позже записываются в state с помошью case метода.

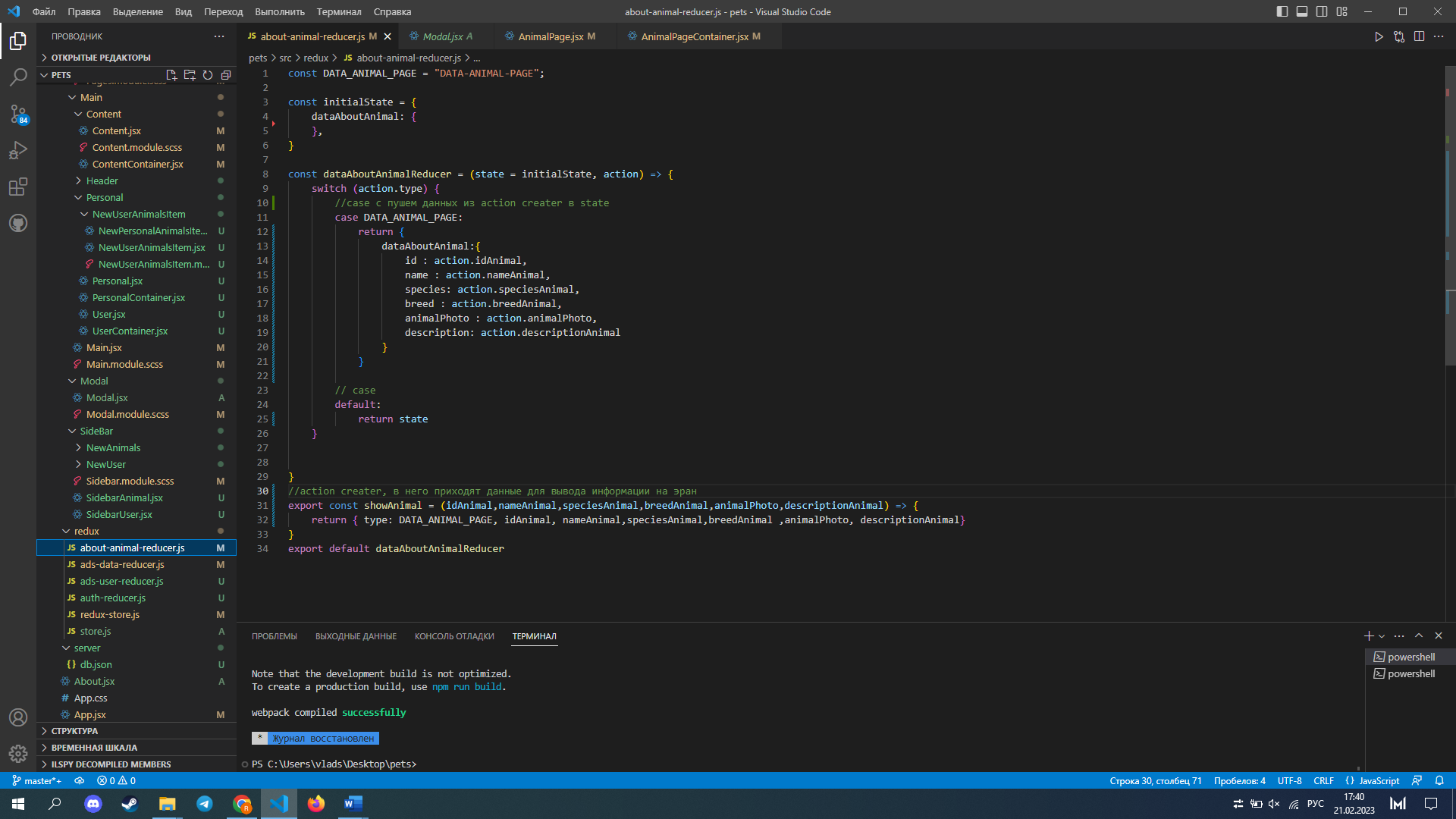


Рисунок 5 –about-animal-reducer

На рисунке 6 предствален json файл с объектами users и animals.

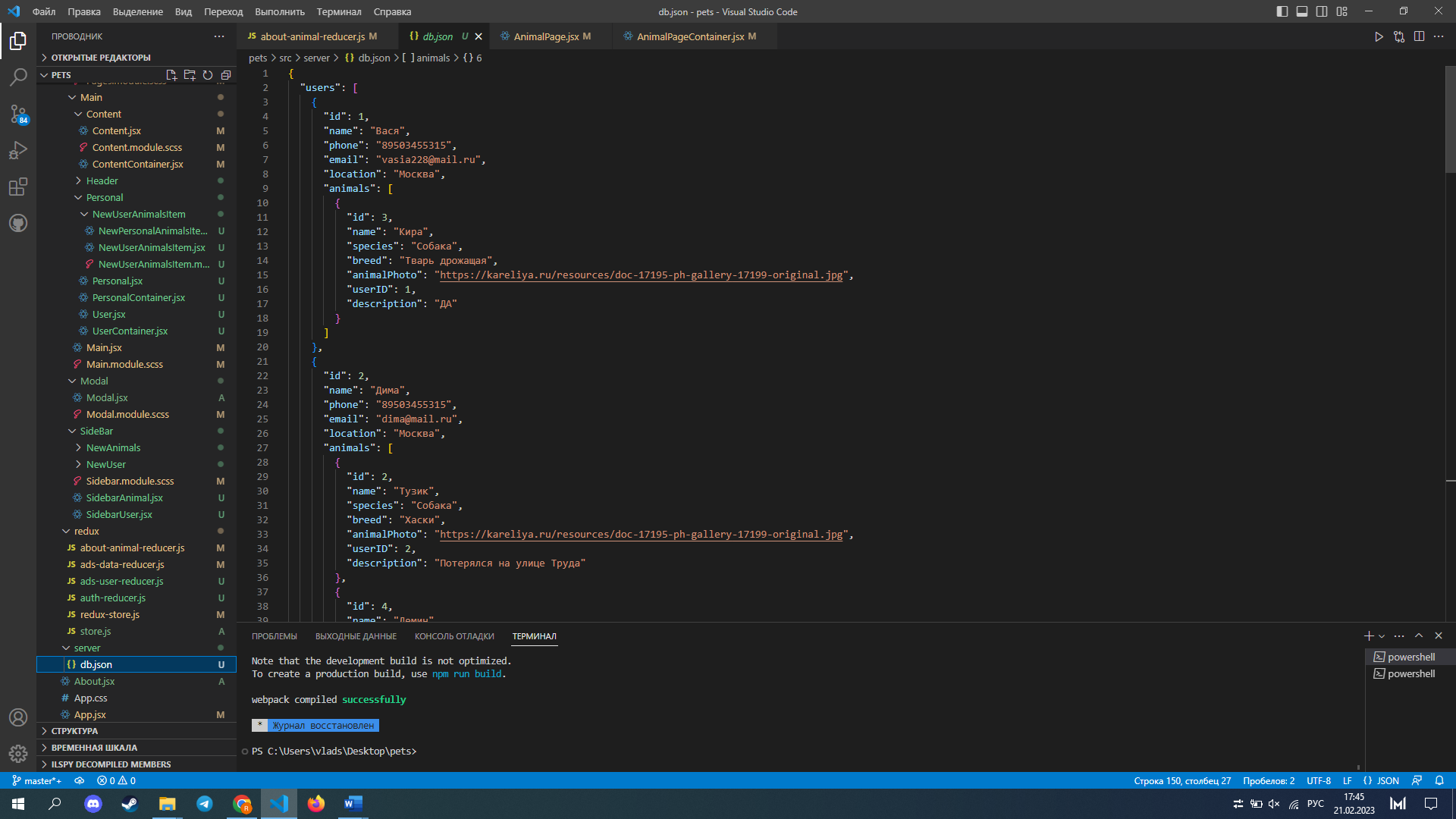
В него записываются данные сформированные при помощи post запросов, а затемм эти данные приложение использует для отрисовки пользователей или животных. 

Рисунок 6 – файл db.json

ЛИСТИНГ ПРОГРАММЫ

**«ads-data-reducer»**

const ADD\_TEXT = "ADD\_TEXT";

const NEW\_ADS\_TEXT = "NEW\_ADS\_TEXT";

const SET\_ANIMALS = "SET\_ANIMALS";

const SET\_CURRENT\_PAGE = "SET\_CURRENT\_PAGE";

const SET\_TOTAL\_ANIMALS = "SET\_TOTAL\_ANIMALS"

const SET\_USERS = "SET\_USERS"

const initialState = {

animals: [],

users: [],

newPostText: {

name: "",

species: "",

breed: "",

animalPhoto: "https://kareliya.ru/resources/doc-17195-ph-gallery-17199-original.jpg",

description: ""

},

pageSize: 4,

totalAnimals: 0,

currentPage: 1,

}

const animalsDataReducer = (state = initialState, action) => {

switch (action.type) {

case ADD\_TEXT:

let idCounter = state.totalAnimals;

idCounter++

let add = {

id: idCounter,

name: action.inputName,

species: action.inputSpecies,

breed: action.inputBreed,

animalPhoto: action.uploadPhoto,

description: action.description

};

return {

...state,

totalAnimals: idCounter,

animals: [...state.animals, add],

newPostText: {

name: "",

species: "",

breed: "",

description: ""

}

}

case NEW\_ADS\_TEXT:

return {

...state,

newPostText: {

name: action.newText,

species: action.newSpeciesText,

breed: action.newBreedText,

}

}

case SET\_ANIMALS: {

return {

...state,

animals: action.animals

}

}

case SET\_CURRENT\_PAGE: {

return {

...state,

currentPage: action.currentPage

}

}

case SET\_TOTAL\_ANIMALS: {

return {

...state,

totalAnimals: action.totalAnimals

}

}

case SET\_USERS: {

return {

...state,

users: action.users

}

}

default:

return state

}

}

export const addText = (inputName, inputSpecies, inputBreed, uploadPhoto, description) => {

return { type: ADD\_TEXT, inputName, inputSpecies, inputBreed, uploadPhoto, description }

}

export const updateText = (newText, newSpeciesText, newBreedText, newDescription) => {

return { type: NEW\_ADS\_TEXT, newText, newSpeciesText, newBreedText, newDescription }

}

export const setAnimals = (animals) => {

return { type: SET\_ANIMALS, animals }

}

export const setCurrentPage = (currentPage) => {

return { type: SET\_CURRENT\_PAGE, currentPage }

}

export const setTotalAnimals = (totalAnimals) => {

return { type: SET\_TOTAL\_ANIMALS, totalAnimals }

}

export const setUsers = (users) => {

return { type: SET\_USERS, users }

}

export default animalsDataReducer