Министерство образования и науки Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №7

по курсу "Разработка web-приложений"  
на тему "Frontend на ReactJS"

Выполнили:

Студенты группы 16ВВ2

Воронков Е.В.

Духанов И.В.

Проверили:

Дубравин А.В.  
Карамышева Н.С.

Пенза 2020

**Цель работы**

Реализовать Frontend на ReactJS.

**Задание**

Необходимо для лабораторной работы №5 реализовать клиентское приложение с использованием технологи ReactJS.

Необходимо в интерфейсе предусмотреть элементы управления для всех параметров, которые можно передать на сервер через REST API. Нужно не забыть, что часть параметров может быть опциональной.

Так же нужно реализовать кнопку, которая должна активировать запрос к серверу и по результатам ответа изменить состояние элементов управления в соответствии с полученными значениями, а также заполнить поле с именем изображения и вместе с этим необходимо выполнить второй запрос к серверу, загрузить изображение по его имени и отобразить на странице.

|  |  |
| --- | --- |
| 2. | Животные: бобер, бизон, бегемот. |

**Листинг**

**Frontend:**

**index.js**

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import App from './App';

import \* as serviceWorker from './serviceWorker';

ReactDOM.render(

<React.StrictMode>

<App />

</React.StrictMode>,

document.getElementById('root')

);

// If you want your app to work offline and load faster, you can change

// unregister() to register() below. Note this comes with some pitfalls.

// Learn more about service workers: https://bit.ly/CRA-PWA

serviceWorker.unregister();

**App.js**

import React, { useState } from 'react';

function App() {

const [name, setName] = useState('')

const [width, setWidth] = useState('')

const [height, setHeight] = useState('')

const [colour, setColour] = useState('')

const [image, setImage] = useState('')

const [img, setImg] = useState('')

const [imgWidth, setImgWidth] = useState('')

const [imgHeight, setImgHeight] = useState('')

/// Функция получения всех параметров

function startSearch(event) {

event.preventDefault()

if (name.trim()) {

let animanlsJSON = {

name: name,

colour: colour,

grouping: document.getElementsByName("grouping")[0].checked ?

document.getElementsByName("grouping")[0].value :

document.getElementsByName("grouping")[1].checked ?

document.getElementsByName("grouping")[1].value : "",

width: width,

height: height,

};

fetch('/api/findimg', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(animanlsJSON)

})

.then(response => {

if(!response.ok) {

throw new Error(response.statusText)

}

return response;

})

.then(response => response.json())

.then(result => {

console.log(result)

if (result.error !== "Image not found")

{

setWidth(result.width)

setHeight(result.height)

setColour(result.animal.colour)

setImage(result.animal.image)

if(result.animal.grouping === "single")

document.getElementsByName("grouping")[0].checked = true;

else

document.getElementsByName("grouping")[1].checked = true;

}

else

setImage('Nothing in DB')

});

}

}

/// Функция получения картинки

function sendReq(event) {

event.preventDefault()

if (name.trim()) {

let animanlsJSON = {

name: name,

colour: colour,

grouping: document.getElementsByName("grouping")[0].checked ?

document.getElementsByName("grouping")[0].value :

document.getElementsByName("grouping")[1].checked ?

document.getElementsByName("grouping")[1].value : "",

};

fetch('/api/img', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(animanlsJSON)

})

.then(response => {

if(!response.ok) {

throw new Error(response.statusText)

}

return response;

})

.then(response => response.json())

.then(result => {

console.log(result)

if (result.image != null)

{

setImgWidth(width)

setImgHeight(height)

setImg('../images/'+ result.image)

}

else

setImage('Image not found')

});

}

}

function reset() {

setName('')

setColour('')

document.getElementsByName("grouping")[0].checked = false;

document.getElementsByName("grouping")[1].checked = false;

setWidth('')

setHeight('')

setImage('')

setImg('')

setImgWidth('')

setImgHeight('')

}

return (

<div className="wrapper">

<label>

Animal Name:

<input

style={{marginLeft: 4}}

value={name}

onChange={event => setName(event.target.value)}

/><p></p>

Colour\*:

<input

style={{marginLeft: 15}}

value={colour}

onChange={event => setColour(event.target.value)}

></input><p></p>

Single\*<input name="grouping" type="radio" value="single"></input>

Multiple\*<input name="grouping" type="radio" value="multiple"></input><br /><br />

Width\*:

<input

style={{marginLeft: 11}}

value={width}

onChange={event => setWidth(event.target.value)}

></input><p></p>

Height\*:

<input

style={{marginLeft: 6}}

value={height}

onChange={event => setHeight(event.target.value)}

></input><p></p>

Image\*:

<input

style={{marginLeft: 10}}

value={image}

onChange={event => setImage(event.target.value)}

></input><p></p>

<input

type="submit"

value="Show Image"

onClick={sendReq}

/>

<input

style={{marginLeft: 20}}

type="submit"

value="Find on server"

onClick={startSearch}

/><p></p>

<input

type="submit"

value="Reset"

onClick={reset}

/> <p></p>

<img

src={img}

width={imgWidth}

height={imgHeight}

onChange={event => setImg(event.target.value)}

/>

</label>

</div>

);

}

export default App;

**Backend:**

**index.js**

const express = require("express");

const mongoose = require("mongoose");

const bodyParser = require("body-parser");

const app = express();

mongoose.connect("mongodb://localhost/animal-db",

{

useNewUrlParser: true,

useUnifiedTopology: true

});

app.use('/images',express.static('images'));

app.use(bodyParser.json());

app.use("/api", require("./api"));

app.listen(8888, () =>{

console.log("server start")

});

**animal.js**

const mongoose = require("mongoose");

const Schema = mongoose.Schema;

const AnimalSchema = new Schema({

name: String,

width: Number,

height: Number,

colour: String,

image: String,

});

const Animal = mongoose.model("animal", AnimalSchema);

module.exports = Animal;

**api.js**

var fs = require("fs");

const express = require("express");

const router = express.Router();

const Animal = require("./animal")

let jsonbody = null;

let imgerror = null;

router.get("/img", (req, res)=>{

console.log(jsonbody);

res.send(jsonbody);

});

router.post("/findimg", (req, res)=>{

console.log("FIND");

console.log(req.body);

if(req.body.colour === ''){

var id = Math.floor(Math.random() \* Math.floor(3));

switch (id) {

case 0:

req.body.colour = "white";

break;

case 1:

req.body.colour = "black";

break;

case 2:

req.body.colour = "purple";

break;

default:

console.log("RandomcolourError");

}

}

if(req.body.grouping === ''){

var id = Math.floor(Math.random() \* Math.floor(2));

switch (id) {

case 0:

req.body.grouping = "single";

break;

case 1:

req.body.grouping = "multiple";

break;

default:

console.log("RandomGroupingError");

}

}

if(!req.body.width){

req.body.width = 0;

while(req.body.width === 0)

req.body.width = Math.floor(Math.random() \* Math.floor(6)) \* 100;

}

if(!req.body.height){

req.body.height = 0;

while(req.body.height === 0)

req.body.height = Math.floor(Math.random() \* Math.floor(6)) \* 100;

}

console.log(req.body);

Animal.findOne({name: req.body.name, colour: req.body.colour, grouping: req.body.grouping})

.then(animal => {

if (animal != null)

{

console.log("FOUND")

console.log({animal: animal, width: req.body.width, height: req.body.height});

res.send({animal: animal, width: req.body.width, height: req.body.height});

}

else

{

imgerror = {error: "Image not found"};

console.log(imgerror.error);

res.send(imgerror);

}

})

});

router.post("/img", (req, res)=>{

console.log("SEND");

console.log(req.body);

Animal.findOne({name: req.body.name, grouping: req.body.grouping, colour: req.body.colour})

.then(animal => {

if (animal != null)

{

console.log("FOUND")

console.log(animal);

res.send(animal);

}

else

{

imgerror = {error: "Image not found"};

console.log(imgerror.error);

res.send(imgerror);

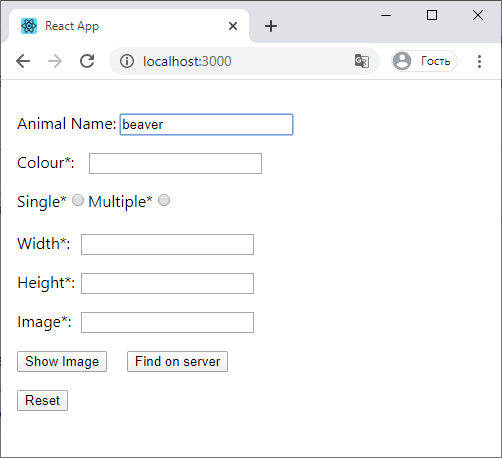
}

});

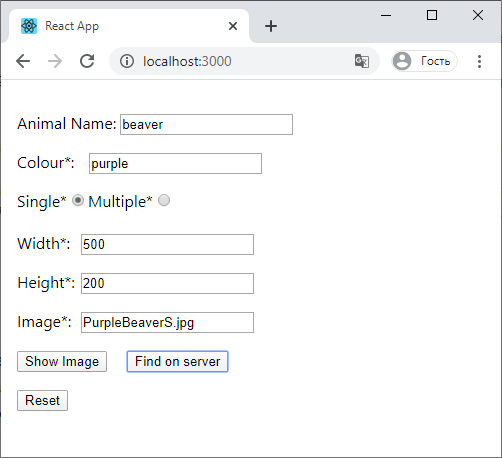
});

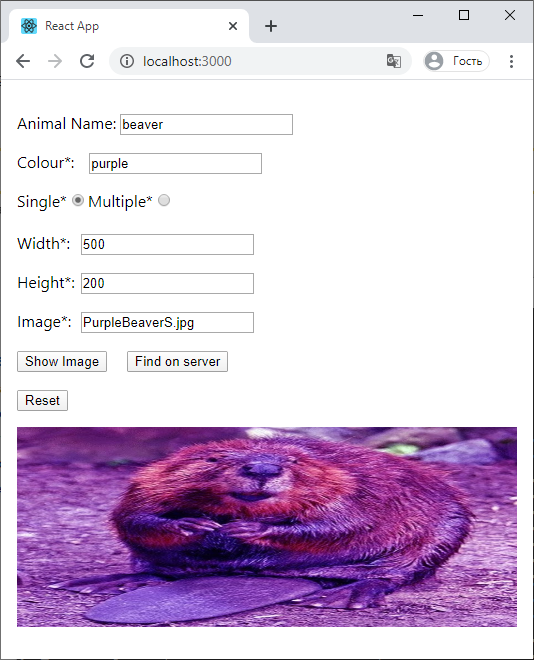
module.exports = router;

**Результат работы**

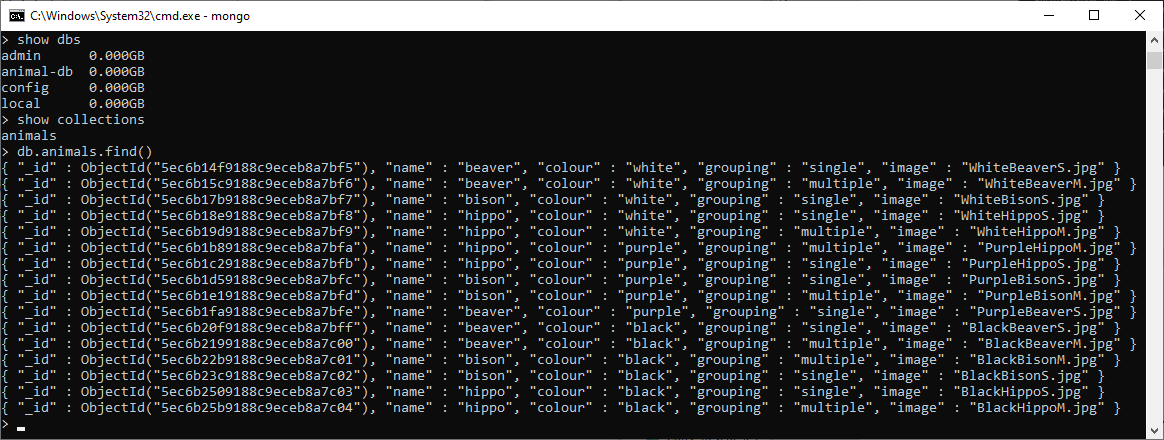


**Рисунок 1 – Введенные значения**

 **Рисунок 2 – Результат работы кнопки “Find on server”**



**Рисунок 2 – Результат работы кнопки “Show Image”**



**Рисунок 3 – Значения в БД**

**Вывод**

Научились реализовывать Frontend на ReactJS.