**ОТЧЕТ**

по лабораторной работе №5

по курсу «Web-программирование»

на тему «Реализация ответа обработчиков и POST-запросы»

Выполнили:

студенты группы 16ВВ1

Кривошеев С.В.

Щетинин К.А.

Принял:

к.т.н., доцент Дубравин А.В.

к.т.н., доцент Карамышева Н.С.

Пенза 2020

Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

### Название

Реализация ответа обработчиков и POST-запросы.

### Цель работы

Реализовать ответ обработчиков и POST-запросы.

### Лабораторное задание

a. На стороне клиента реализовать проверку: если в текстовом поле указано одно из трех животных в следующем формате:«antilopa»или «antilopa 7»(число от 0 до 9), выполнить формирование и отправку на сервер POST-запроса, в противном случае вывести сообщение о некорректно указанных данных.

b. На стороне сервера реализовать три обработчика, соответствующих каждому из указанных в варианте животных.

c. В обработчике, в случае, если в URLзапросе отсутствует параметр с указанием номера запрашиваемого изображения животного, выполнить генерацию случайного числа в диапазоне от 0 до 9 и выполнить отправку клиенту соответствующего изображения. Если в URLзапросе присутствует параметр с указанием номера запрашиваемого изображения животного – выполнить отправку клиенту соответствующего изображения.

|  |  |
| --- | --- |
| Вариант | Задание |
| 4 | Животные: гиена, газель, геккон. |

### Порядок выполнения работы

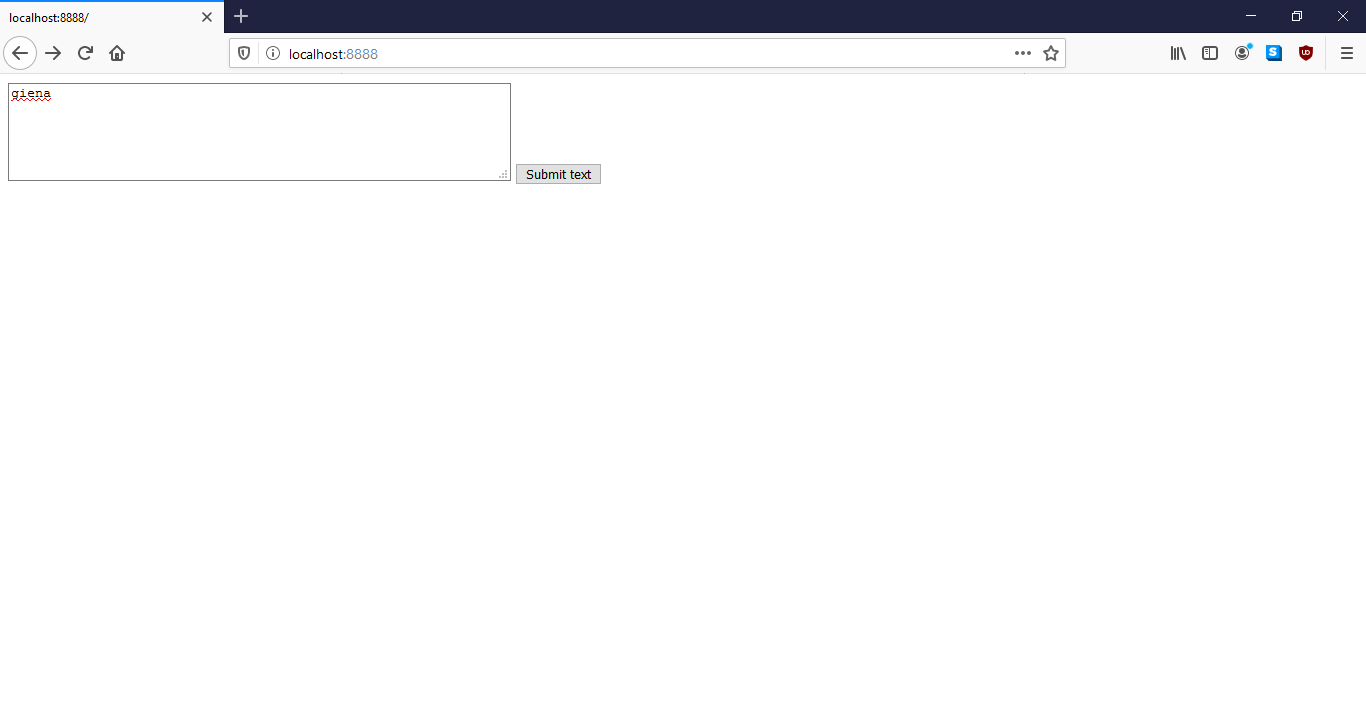


Рисунок 1 – ввод сообщения

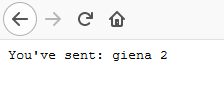


Рисунок 2 - вывод сообщения без указания числа



Рисунок 3 - вывод в консоль



Рисунок 4 - ввод сообщения с числом

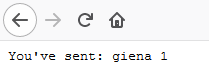


Рисунок 5 - вывод сообщения с числом



Рисунок 6 - вывод в консоль

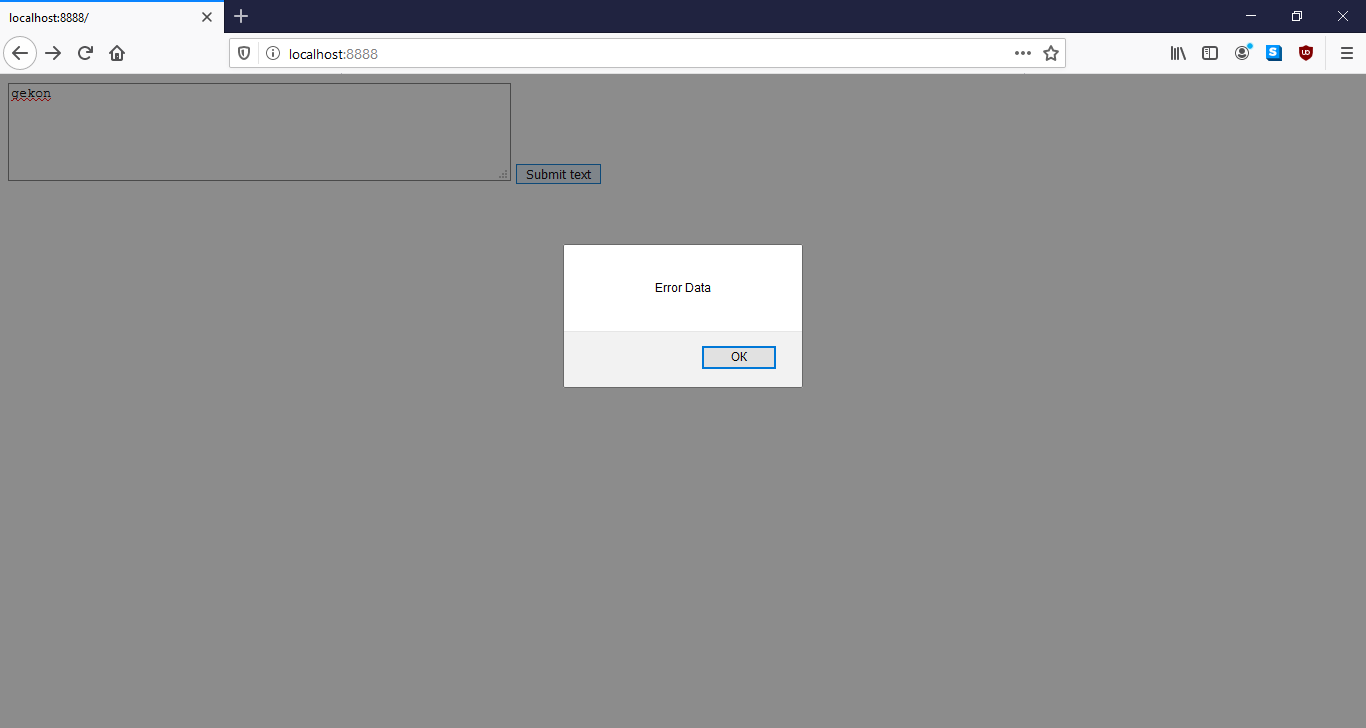


Рисунок 7 – проверка и вывод ошибки на стороне клиента

### Листинг

#### Five.html

<html>

<head>

<meta http-equiv="Content-Type" content="text/html;

charset=UTF-8" />

<script>

let animanls = ["giena", "gazel", "gekkon"];

function validate(form) {

for (let i = 0; i < animanls.length; i++) {

if (form.text.value == animanls[i]) {

return true;

}

for (let j = 0; j < 9; j++) {

if (form.text.value == animanls[i] + " " + j) {

return true;

}

}

}

alert("Error Data");

return false;

}

</script>

</head>

<body>

<form method="post" onsubmit="return validate(this);">

<textarea name="text" rows="5" cols="60"></textarea>

<input type="submit" value="Submit text" />

</form>

</body>

</html>

#### Server.JS

var fs = require("fs");

var http = require("http");

var url = require("url");

var querystring = require("querystring");

function start(route, handle) {

function onRequest(request, response) {

var postData = "";

var pathname = url.parse(request.url).pathname;

console.log("Request for " + pathname + " received.");

request.setEncoding("utf8");

request.addListener("data", function(postDataChunk) {

postData += postDataChunk;

pathname ="/"+ querystring.parse(postData).text.split(' ')[0];

console.log("Received POST data chunk '" +

postDataChunk + "'.");

});

request.addListener("end", function() {

route(handle, pathname, response, postData);

});

}

http.createServer(onRequest).listen(8888);

console.log("Server has started");

}

exports.start = start;

#### Index.JS

var server = require("./server");

var router = require("./router");

var requestHandlers = require("./requestHandlers");

var handle = {}

handle["/"] = requestHandlers.start;

handle["/start"] = requestHandlers.start;

handle["/upload"] = requestHandlers.upload;

handle["/download"] = requestHandlers.download;

handle["/giena"] = requestHandlers.giena;

handle["/gazel"] = requestHandlers.gazel;

handle["/gekkon"] = requestHandlers.gekkon;

server.start(router.route, handle);

#### router.JS

function route(handle, pathname, response, postData) {

console.log("About to route a request for " + pathname);

if (typeof handle[pathname] === 'function') {

handle[pathname](response, postData);

} else {

console.log("No request handler found for " + pathname);

response.writeHead(404, {

"Content-Type": "text/plain"

});

response.write("404 Not found");

response.end();

}

}

exports.route = route;

#### requestHandlers.JS

var exec = require("child\_process").exec;

var fs = require("fs");

var querystring = require("querystring");

function start(response, postData) {

console.log("Был вызван обработчик запроса START");

var body = fs.readFileSync('five.html');

response.writeHead(200, {

"Content-Type": "text/html"

});

response.write(body);

response.end();

}

function raspred(response, postData){

var pdata = querystring.parse(postData).text.split(' ')[0];

var chdata = querystring.parse(postData).text.split(' ')[1];

response.writeHead(200, {

"Content-Type": "text/plain"

});

if (chdata != null) {

response.write("You've sent: " + querystring.parse(postData).text);

response.end();

} else {

response.write("You've sent: " + querystring.parse(postData).text + " " + Math.floor(Math.random() \* 10));

response.end();

}

}

function upload(response, postData) {

console.log("Был вызван обработчик запроса UPLOAD");

raspred(response, postData)

}

function download() {

console.log("Был вызван обработчик запроса DOWNLOAD");

}

function giena(response, postData) {

console.log("Был вызван обработчик запроса GIENA");

raspred(response, postData)

}

function gazel(response, postData) {

console.log("Был вызван обработчик запроса GAZEL");

raspred(response, postData)

}

function gekkon(response, postData) {

console.log("Был вызван обработчик запроса GEKKON");

raspred(response, postData)

}

exports.start = start;

exports.upload = upload;

exports.download = download;

exports.giena = giena;

exports.gazel = gazel;

exports.gekkon = gekkon;

### Выводы

В ходе выполнения лабораторной работы реализованы ответы обработчиков и POST-запросы.