Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа № 3

по дисциплине «Проектирование и

разработка веб-приложений»

Выполнил:

ст. гр. ИРсп-121

Орлова С. В.

Приняла:

Максимова А. С.

Владимир 2023 г.

**Цель работы**

создание HTTP сервера с помощью express, добавление модульных обработчиков express, обработка ошибок express, обработка GET и POST запросов.

**Задание для выполнения:**

1. Изучить теорию в методичке;

2. Разработать собственный веб-сервер, в котором будут

поддерживаться 4 типа запросов (GET, POST, PUT, DELETE) по варианту. № варианта = № в журнале.

(пояснение 2 задания) Реализовать возможность получения массива данных с помощью GET-запроса, получение определённого элемента по идентификатору, добавление элемента с помощью POST запроса, редактирование с помощью PUT-запроса и удаление с помощью DELETE запроса. (Всего 5 запросов). Добавить обработку исключительных случаев (например, если объект не найден). Добавьте файл.

3. Обработайте ошибки с помощью Express.

4. Продемонстрировать работу веб-сервера с помощью Postman.

5. Составить отчет по результатам работы

**Практическая часть**

**Вариант 1**

Веб-сервер для работы с пользователями (добавить пользователя, вывести список пользователей, удалить пользователя, обновление данных пользователя).

**Выведен список пользователей**

**Листинг файла UserController.js:**

var express = require('express');

var router = express.Router();

class User {

login;

email;

constructor(login, email) {

this.email = email;

this.login = login;

}

}

let users = [{ id: 1, name: 'Евгений' },

{ id: 2, name: 'Мария' },

{ id: 3, name: 'Артем' }

];

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

res.send(users)

});

module.exports = router;

Листинг файла Index.js:

const express = require('express'), http = require('http');

const usersController = require("./UserController.js");

const app = express();

const server = http.createServer(app);

app.use("/users", usersController);

app.use("",(req, res) => {

res.send("Работает")

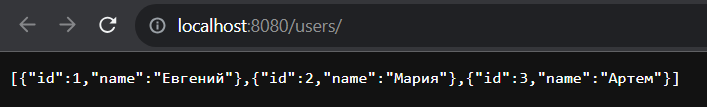
})

const port = 8080

server.listen(port, () => {

console.log('Server listening on port: ', port);

})



**Добавлен файл**

Листинг файла UserController.js:

var multer = require('multer');

var upload = multer();

router.post("",

upload.any(),

(req, res) => {

console.log(req.files)

console.log(req.body)

res.send('Все прошло успешно')

})

Были ознакомлены с материалом методического указания.

Например, напишем код для файла Index.js.

**Листинг:**

const express = require('express'), http = require('http');

const usersController = require("./controller/UserController");

const app = express();

const server = http.createServer(app);

app.use("/users", usersController);

app.use("",(req, res) => {

res.send("Работает")

})

const port = 8083

server.listen(port, () => {

console.log('Server listening on port: ', port);

})

Добавим отдельный пакет controller, в котором добавим UserController.js

**Листинг:**

var express = require('express');

var router = express.Router();

class User {

login;

email;

constructor(login, email) {

this.email = email;

this.login = login;

}

}

let users = [new User("Пользователь", "123@vlsu.ru")]

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

res.send(users)

});

module.exports = router;

Потом запустим его командой: node Index.js



Рисунок1 – Результат

Добавим поддержку json в корневом файле.

**Листинг:**

app.use(express.json())

Добавляем в файл UserController.js пару строк кода.

**Листинг:**

var multer = require('multer');

var upload = multer();

router.post("",

upload.any(),

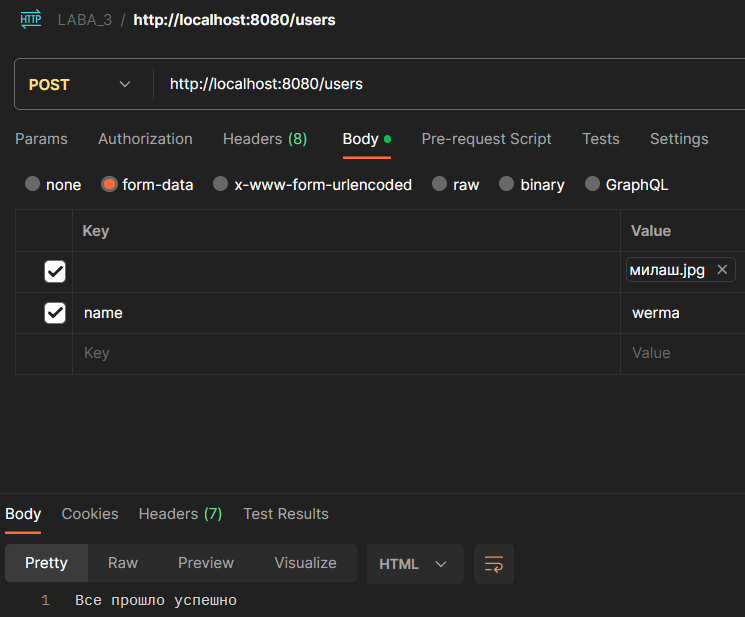
(req, res) => {

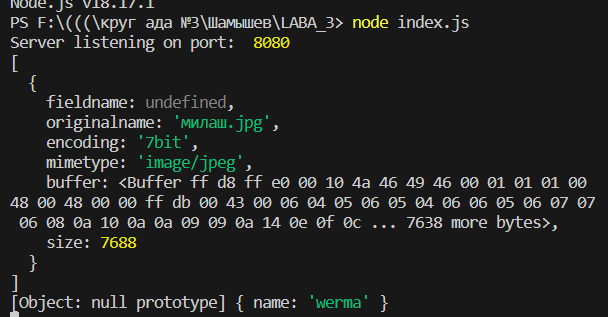
console.log(req.files)

console.log(req.body)

res.send('Все прошло успешно')

})





Теперь можно обработать тип raw(JSON). Представлен код.

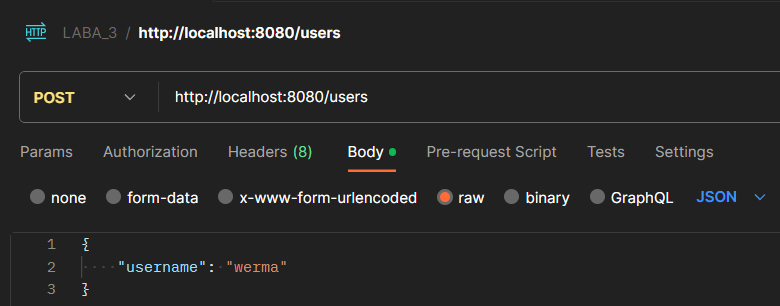
**Листинг:**

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

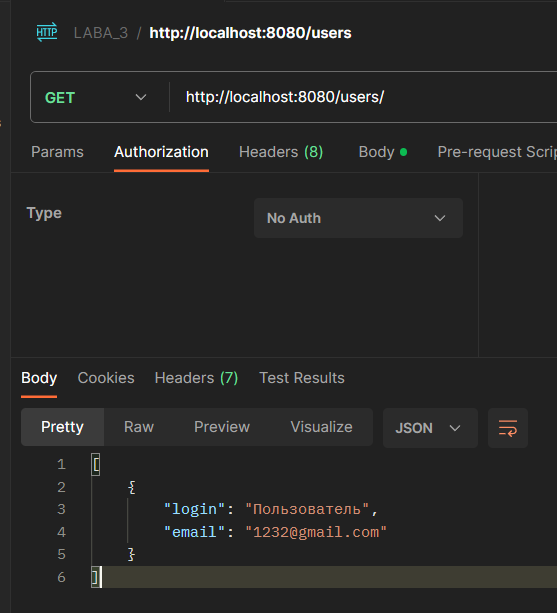
console.log(req.body);

res.send(req.body)

})







**Задание 2**

Разработать собственный веб-сервер, в котором будут поддерживаться 4 типа запросов (GET, POST, PUT, DELETE) по варианту. № варианта = № в журнале.

(пояснение 2 задания) Реализовать возможность получения массива данных с помощью GET-запроса, получение определённого элемента по идентификатору, добавление элемента с помощью POST запроса, редактирование с помощью PUT-запроса и удаление с помощью DELETE запроса. (Всего 5 запросов). Добавить обработку исключительных случаев (например, если объект не найден). Добавьте файл.

**GET-запрос. Получение определённого элемента по идентификатору**

**Листинг:**

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

res.send(users[req.params.userId]);

id = req.params.userId;

if (id > users.length-1) {

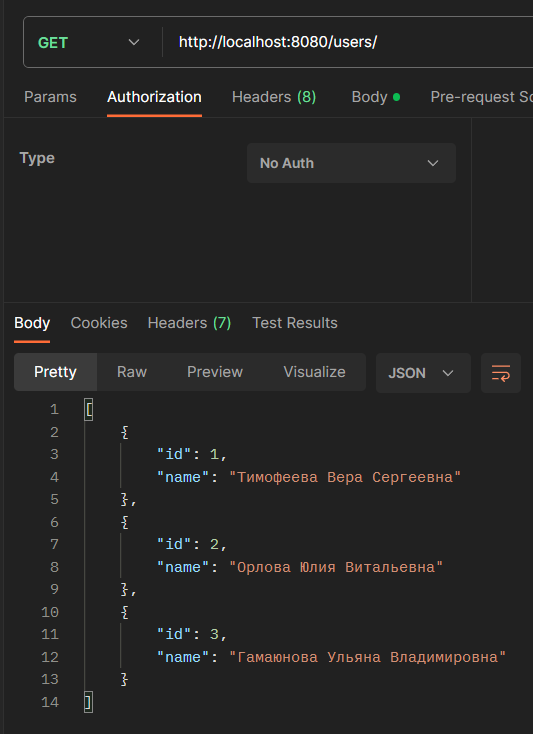
res.status("404")

res.send({message: "Неизвестный"})

}

res.send(users[req.params.userId]);

})



**Добавление элемента с помощью POST запроса**

**Листинг:**

var multer = require('multer');

var upload = multer();

router.post("",

upload.any(),

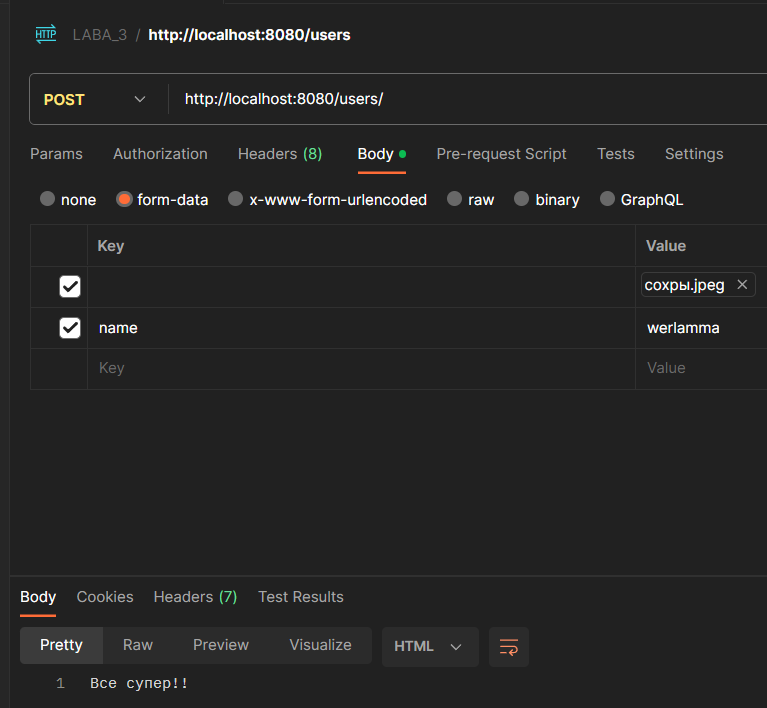
(req, res) => {

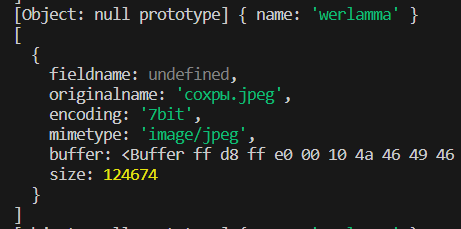
console.log(req.files)

console.log(req.body)

res.send('Все супер!!')

})





**PUT**

DELETE