**ОТЧЕТ**

по лабораторной работе №3

по курсу «Web-программирование»

на тему «Формирование собственных модулей и реализация обработки GET-запросов»

Выполнили:

студенты группы 16ВВ1

Соловьев А.А.

Чиркин К.Д.

Принял:

к.т.н., доцент Дубравин А.В.

Пенза 2020

Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

### Лабораторное задание.

1. Выполнить рефакторинг кода ранее созданного http-сервера.
2. Реализовать обработку входящих http-запросов следующим образом:
3. Выполнить получение двух параметров из URL входящего запроса.
4. Выполнить суммирование числовых значений параметров.
5. Выполнить передачу ответа клиенту.

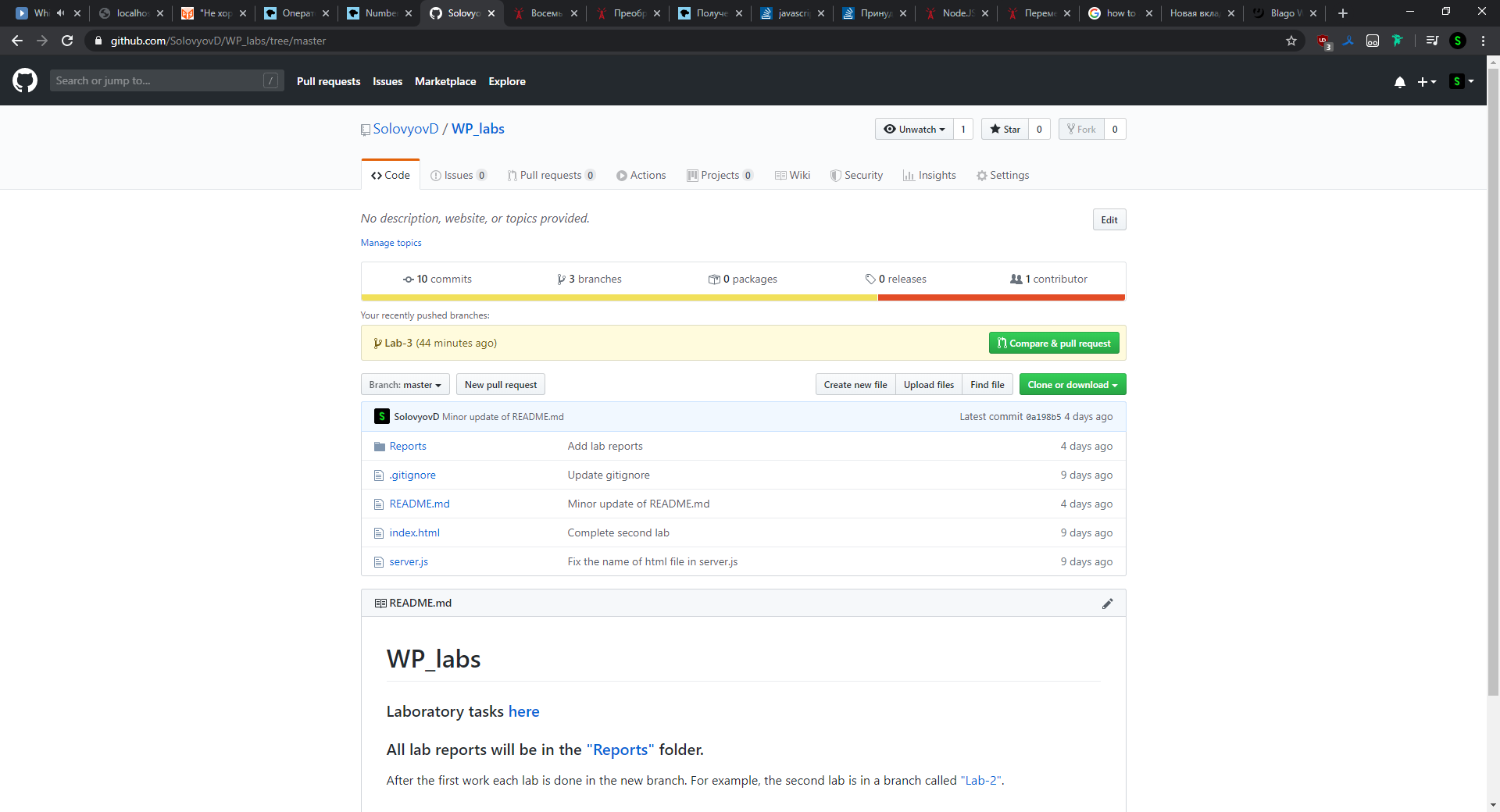
* Если значения параметров представлены не числовыми значениями, в качестве ответа клиенту вывести «Error!

### Ход работы.

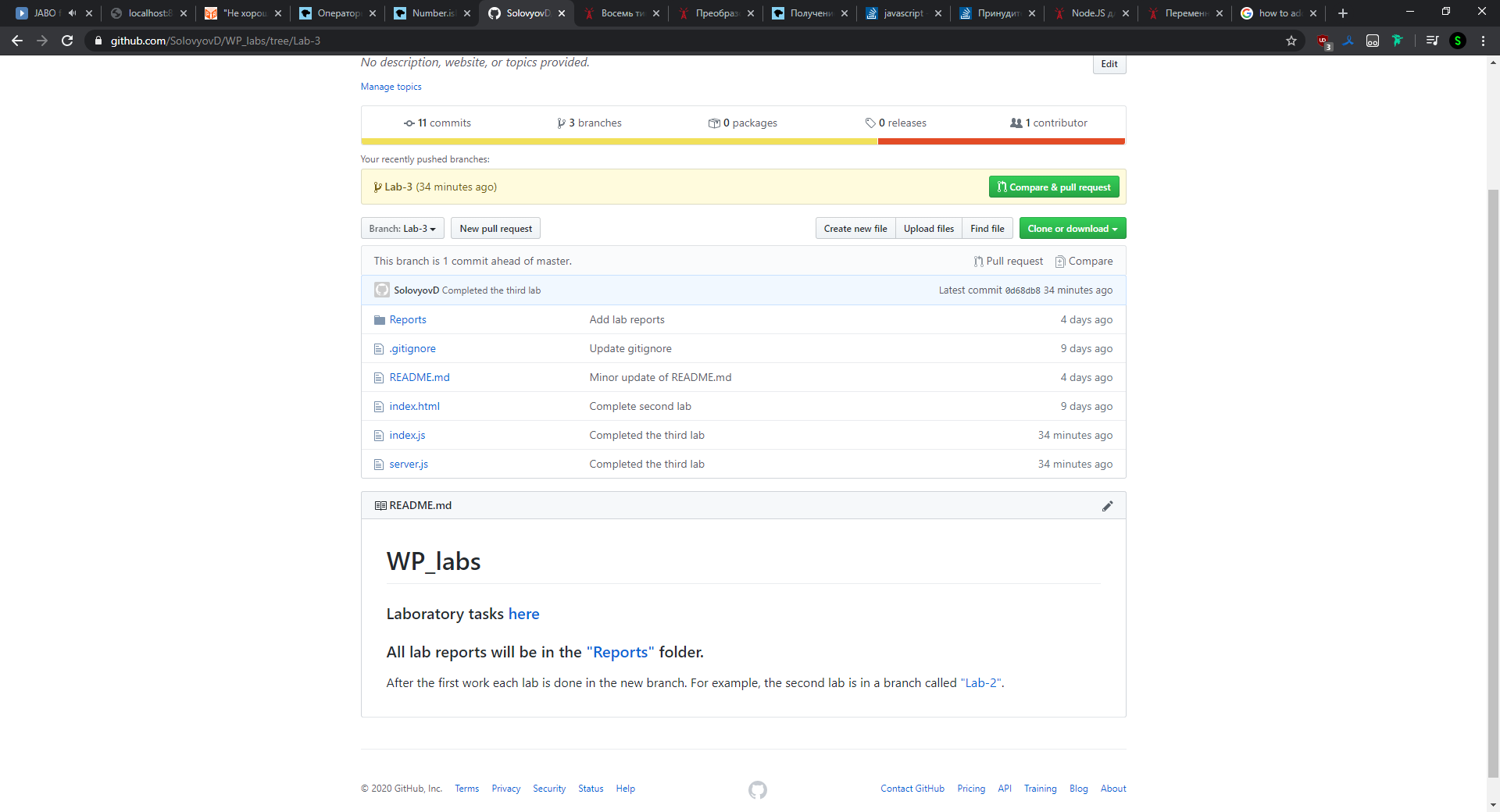
#### Рефакторинг http-сервера.

Общепринято иметь главный файл с названием index.js, который используется для начальной загрузки и запуска приложения, путём использования других модулей приложения.

Поэтому заносим код сервера под функцию start в файле server.js, экспортируем и вызываем в index.js. Структура проекта принимает следующий вид:



↓



**Рисунок 1. Структура проекта.**

#### Реализация обработки входящих http-запросов.

Вся необходимая информация доступна через объект request, который передается в качестве первого параметра callback-функции onRequest(). Чтобы интерпретировать эту информацию, необходимо подключить дополнительные Node.js-модули, а именно url и querystring. Модуль url поддерживает методы, которые позволяют извлекать различные части URL (такие как запрошенный путь (URL path) и строка параметров запроса (search)), а querystring в свою очередь, используется для парсинга строки параметров запроса.

Используем возможности приведенных модулей для выполнения лабораторного задания. В итоге содержимое server.js следующее:

const fs = require("fs");

const http = require("http");

const url = require('url');

const querystring = require('querystring');

function start() {

    function onRequest(request, response) {

        console.log("Request received");

        var sum = 0;

        var sp = url.parse(request.url).search;

        if (sp != null) {

            var parsed\_sp = querystring.parse(sp.split('?')[1]);

            for (key in parsed\_sp) {

                if (Number.isInteger(Number(parsed\_sp[key]))) {

                    sum += Number(parsed\_sp[key]);

                }

            }

            if(sum == 0){sum = "Error!";}

            console.log("Result = " + sum);

            response.write(sum.toString());

            response.end();

            return;

        }

        var page = fs.readFileSync('index.html');

        response.writeHead(200, { 'Content-Type': 'text/html' });

        response.write(page);

        response.end();

    }

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

    console.log("Server has started");

}

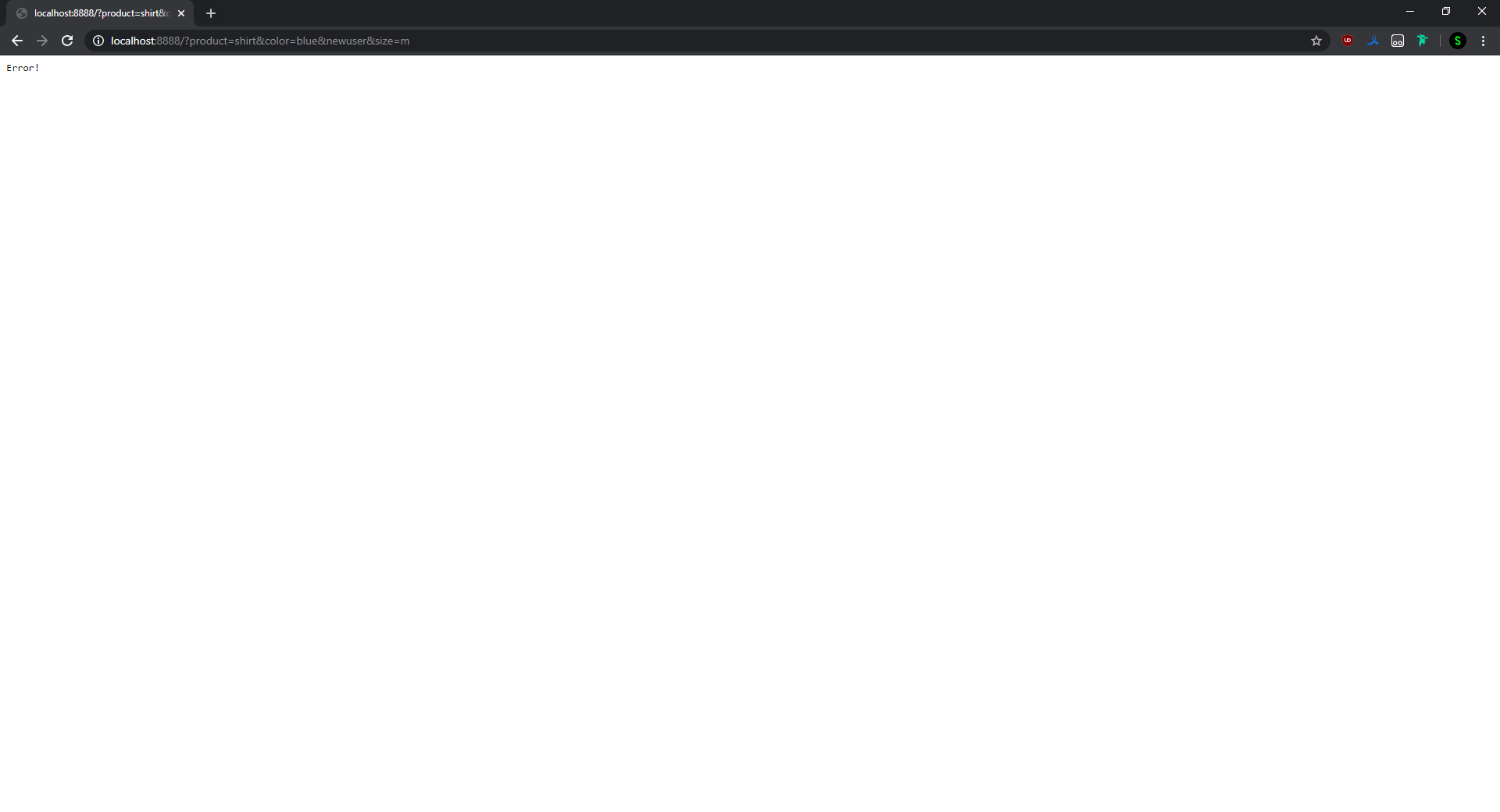
exports.start = start;

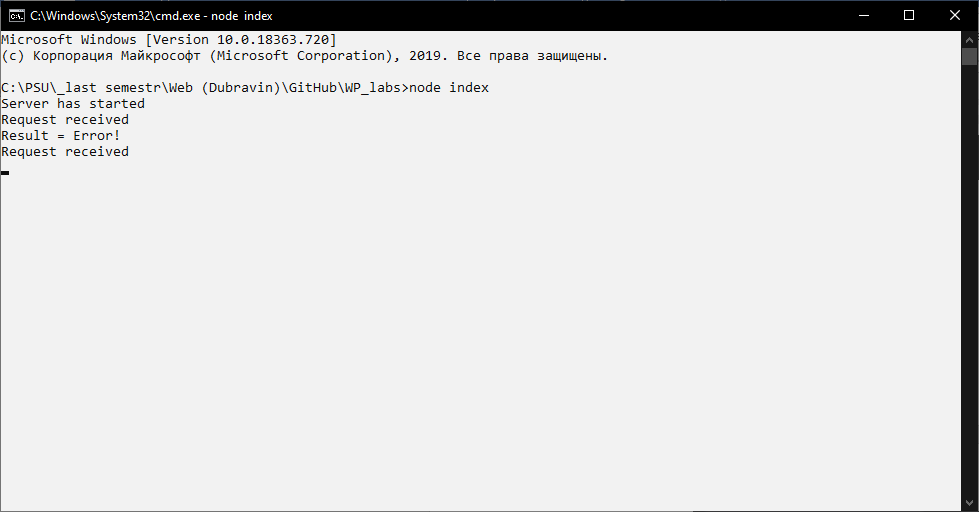
Содержимое index.js:

var server = require("./server");

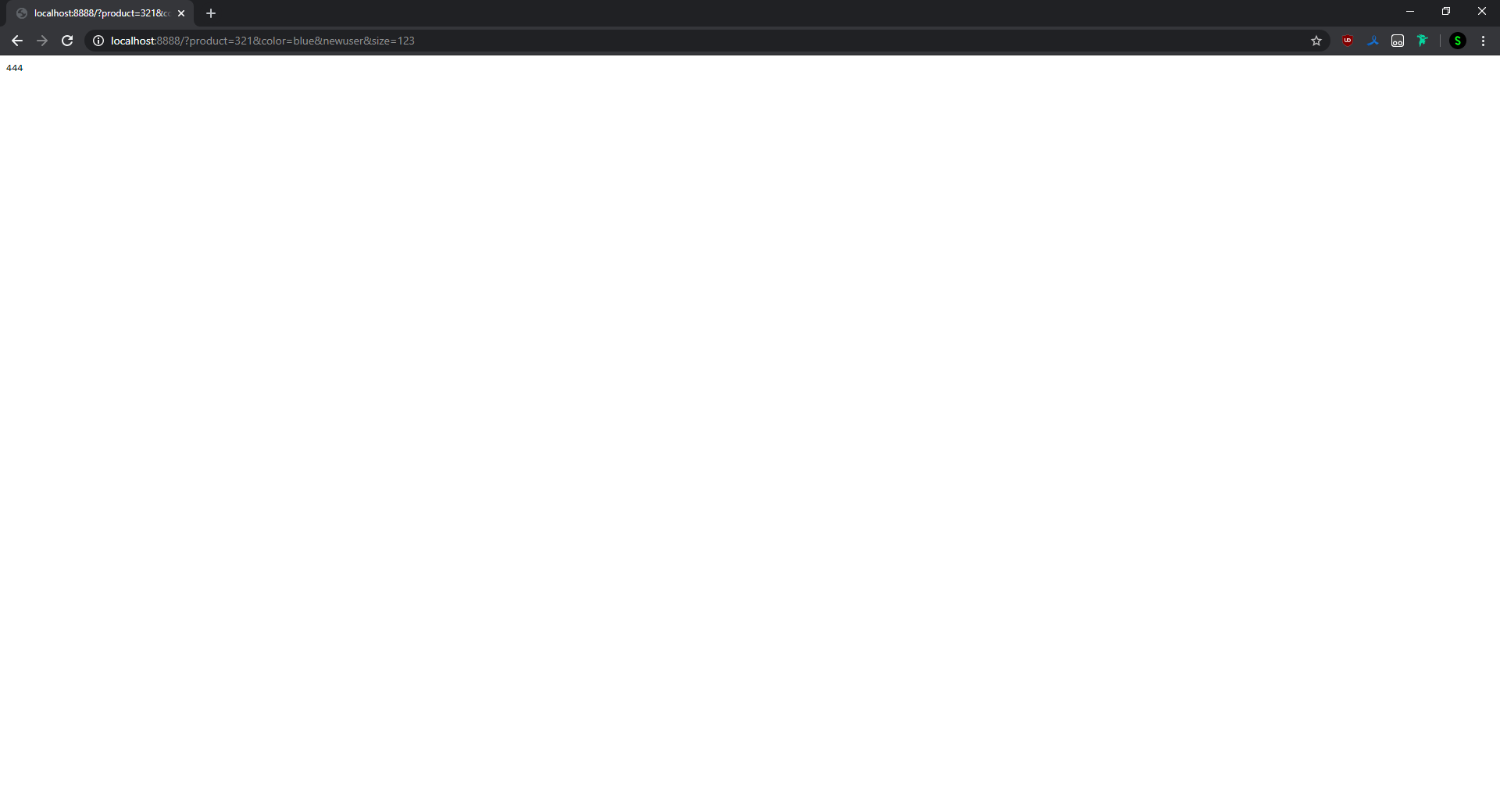
server.start()

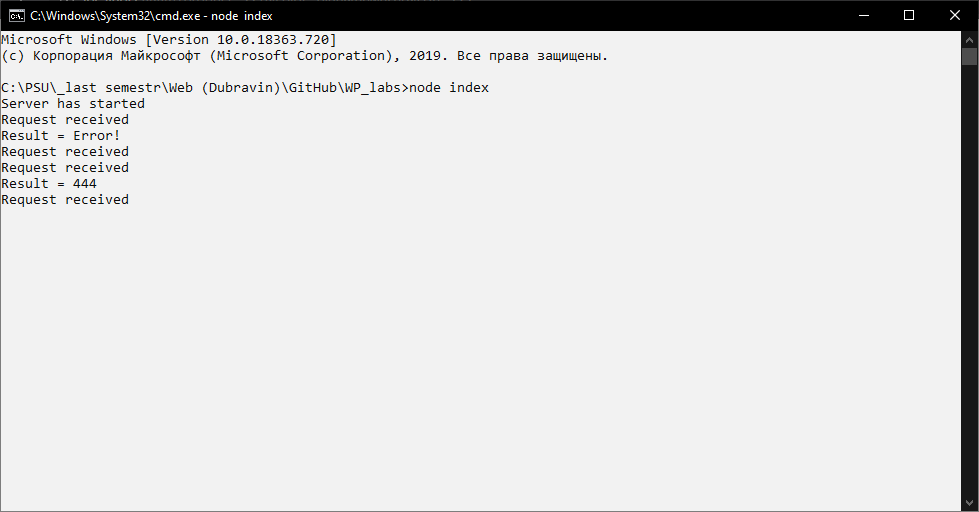
Проверяем работоспособность:





**Рисунок 2. Вывод при отсутствии числовых значений в запросе.**





**Рисунок 3. Результат при наличии числовых значений в запросе.**

### Выводы

В ходе данной лабораторной работы получили практические навыки обработки входящих http-запросов.