|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**ОТЧЕТ**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 3 |

**Название:**

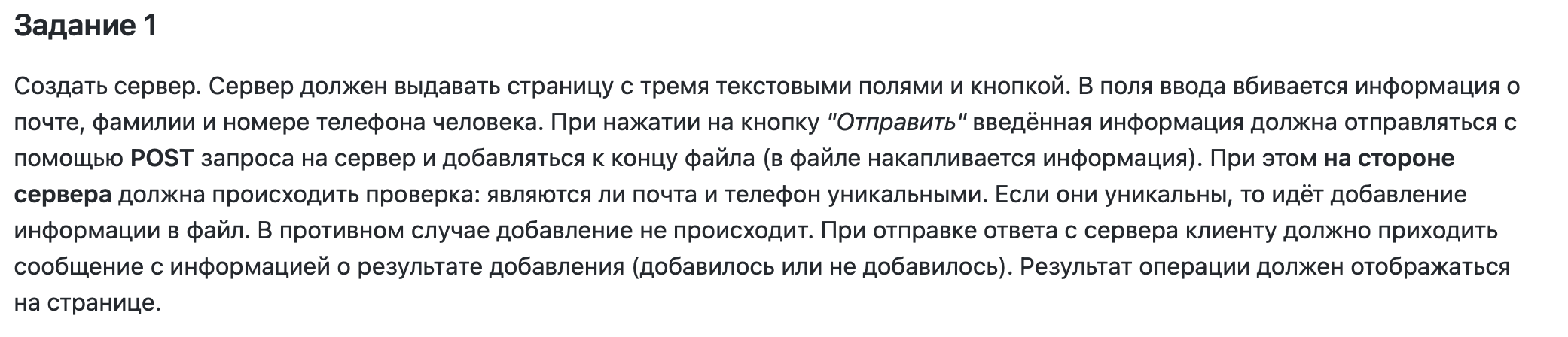
POST, GET, CSS, URL, COOKIE

**Дисциплина:** Архитектура ЭВМ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-52Б |  |  | Блохин Д. М. |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | А.Ю. Попов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

TASK 5-6.



"use strict";

const { RSA\_NO\_PADDING } = require("constants");

const e = require("express");

const express = require("express");

const fs = require("fs");

const app = express();

const port = 5000;

app.listen(port);

console.log(`Server on port ${port}`);

const way = \_\_dirname + "/static";

app.use(express.static(way));

app.use(function(req, res, next) {

res.header("Cache-Control", "no-cache, no-store, must-revalidate");

res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");

res.header("Access-Control-Allow-Origin", "\*");

next();

});

app.get("/sum", function(request, response) {

const a = request.query.a;

const b = request.query.b;

const s = parseInt(a) + parseInt(b);

response.end(JSON.stringify({

result: s

}));

});

function getPersonByEmail(email, arr) {

let reqIndex = -1;

for (let i = 0; i < arr.length && reqIndex === -1; i++) {

if (arr[i]["email"] === email) {

reqIndex = i;

}

}

if (reqIndex === -1) {

return null

}

return arr[reqIndex];

}

app.get("/person", function(request, response) {

const email = request.query.email;

let res = "";

const fileName = "task1.txt";

if (!fs.existsSync(fileName)) {

res = "There is no such email.";

}

else {

const contentString = fs.readFileSync(fileName, "utf-8");

const arr = JSON.parse(contentString);

const reqObj = getPersonByEmail(email, arr);

if (!reqObj) {

res = "There is no such email.";

}

else {

res = JSON.stringify(reqObj);

}

}

response.end(JSON.stringify({

result: res

}));

})

function loadBody(request, callback) {

let body = [];

request.on('data', (chunk) => {

body.push(chunk);

}).on('end', () => {

body = Buffer.concat(body).toString();

callback(body);

});

}

function isUniqueNumberAndEmail(obj, arr) {

let res = true;

for (let i = 0; i < arr.length && res; i++) {

if (obj["email"] === arr[i]["email"] ||

obj["phoneNumber"] === arr[i]["phoneNumber"]) {

res = false;

}

}

return res

}

app.post("/save/info", function(request, response) {

loadBody(request, function(body) {

const fileName = "task1.txt";

if (!fs.existsSync(fileName)) {

fs.writeFileSync(fileName, "[]");

}

const obj = JSON.parse(body);

const prevContentString = fs.readFileSync(fileName, "utf-8");

const arr = JSON.parse(prevContentString);

let resultMessage = "Success";

if (isUniqueNumberAndEmail(obj, arr)) {

arr.push(obj);

const contentString = JSON.stringify(arr, null, 2);

fs.writeFileSync("task1.txt", contentString);

}

else {

resultMessage = "Not Unique";

}

response.end(JSON.stringify({

result: resultMessage

}));

});

});

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Add Email</title>

<link rel="stylesheet" href="/style1.css">

</head>

<body>

<h1>Add Email</h1>

<p>Email:</p>

<input id="field-email" type="text" spellcheck="false" autocomplete="off">

<p>Surname:</p>

<input id="field-surname" type="text" spellcheck="false" autocomplete="off">

<p>Phone Numbe:r</p>

<input id="field-phone" type="text" spellcheck="false" autocomplete="off">

<br>

<br>

<div onclick="makeAction()" class="btn-class">Send</div>

<script>

"use strict";

function ajaxPost(urlString, bodyString, callback) {

let r = new XMLHttpRequest();

r.open("POST", urlString, true);

r.setRequestHeader("Content-Type", "application/json;charset=UTF-8");

r.send(bodyString);

r.onload = function() {

callback(r.response);

}

}

function makeAction() {

const email = document.getElementById("field-email").value;

const surname = document.getElementById("field-surname").value;

const phoneNumber = document.getElementById("field-phone").value;

ajaxPost("/save/info", JSON.stringify({

email, surname, phoneNumber

}), function(answerString) {

const answerObject = JSON.parse(answerString);

const result = answerObject.result;

alert(result);

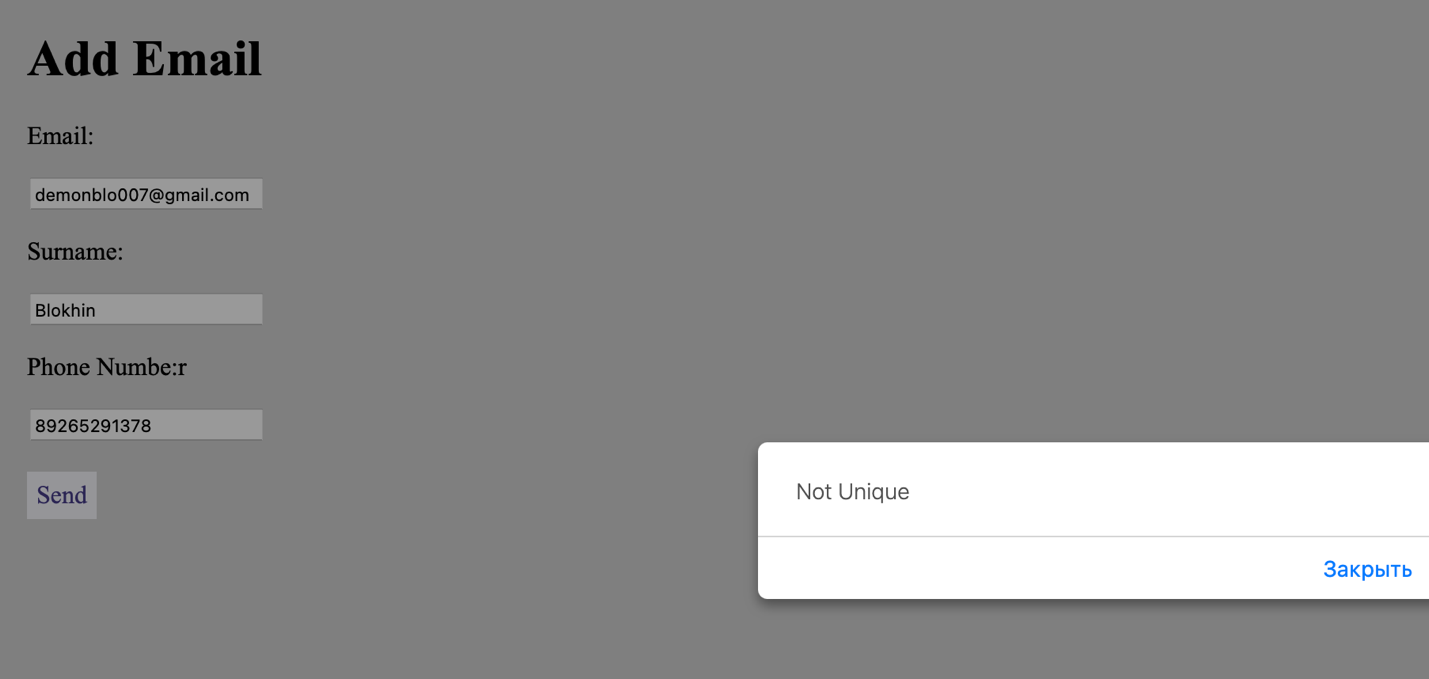
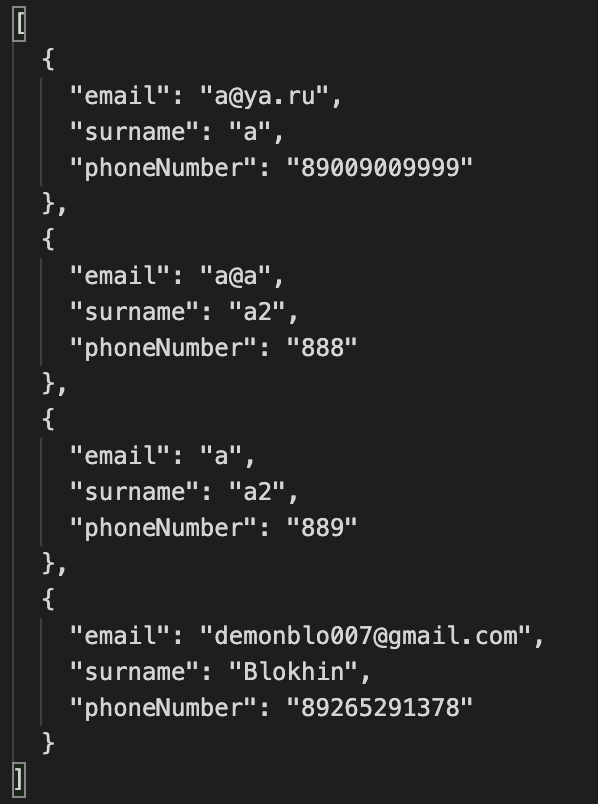
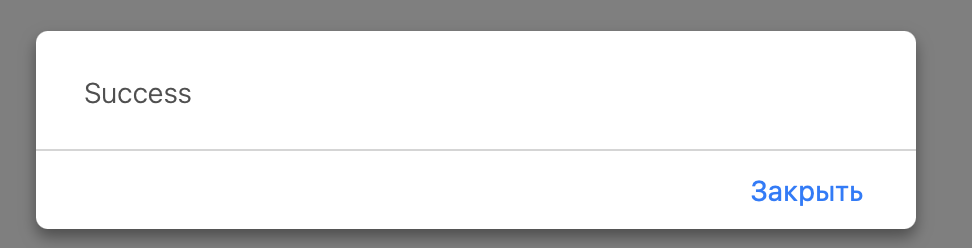
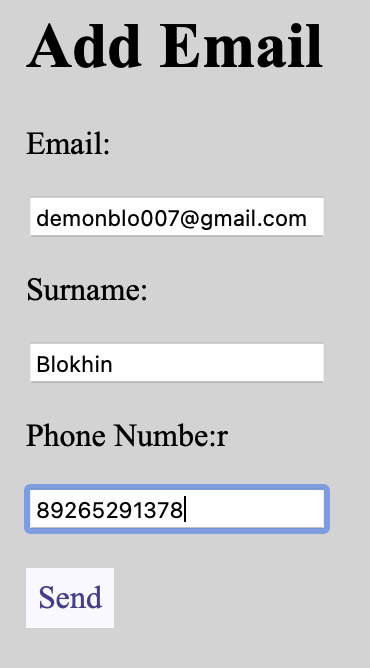
});

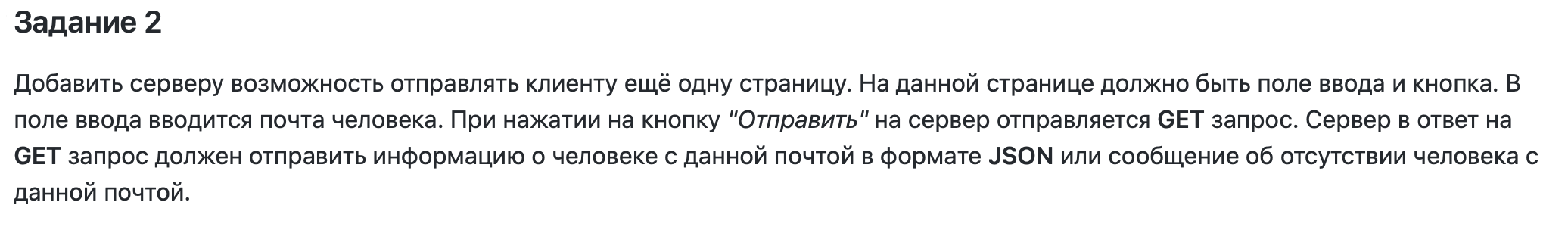
}

</script>

</body>

</html>





<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Result</title>

<link rel="stylesheet" href="/style2.css">

</head>

<body>

<h1>Find email</h1>

<p>Enter search:</p>

<input class="input-class" id="field-email" type="text" spellcheck="false" autocomplete="off">

<br>

<br>

<div id="get-btn" class="btn-class">Отправить</div>

<br>

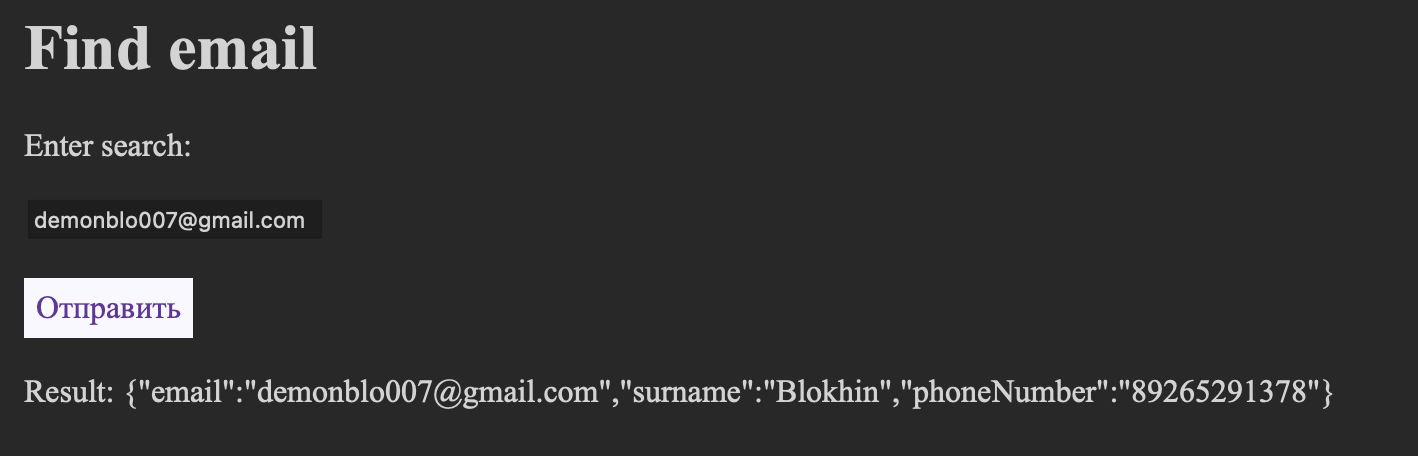
<br>

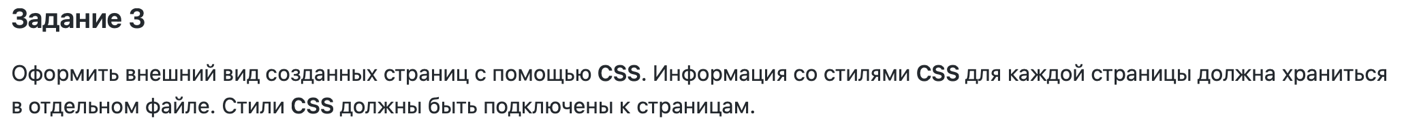
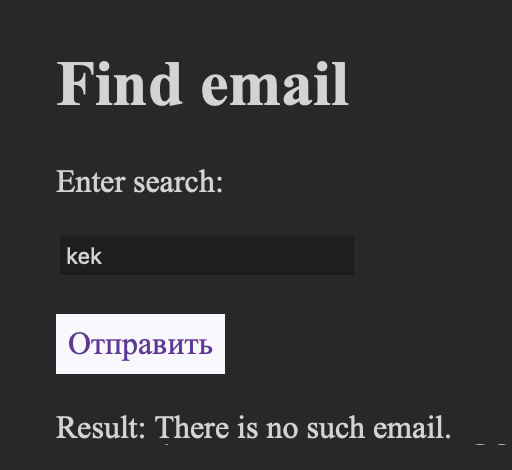
<p1 id="result-label"></p1>

<script src="/code2.js"></script>

</body>

</html>





Задание 1:

body {

padding: 30px;

background: lightgray;

font-family: 'Times New Roman', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif

}

.btn-class {

padding: 6px;

background: ghostwhite;

color: darkslateblue;

cursor: pointer;

display: inline-block;

}

Задание 2:

body {

padding: 30px;

background: rgb(40, 40, 40);

color: lightgray;

font-family: 'Times New Roman', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva, Verdana, sans-serif

}

.btn-class {

padding: 6px;

background: ghostwhite;

color: rebeccapurple;

cursor: pointer;

display: inline-block;

}

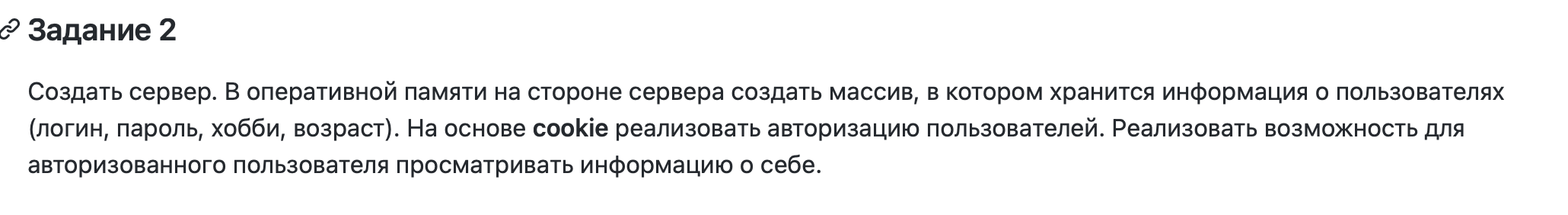
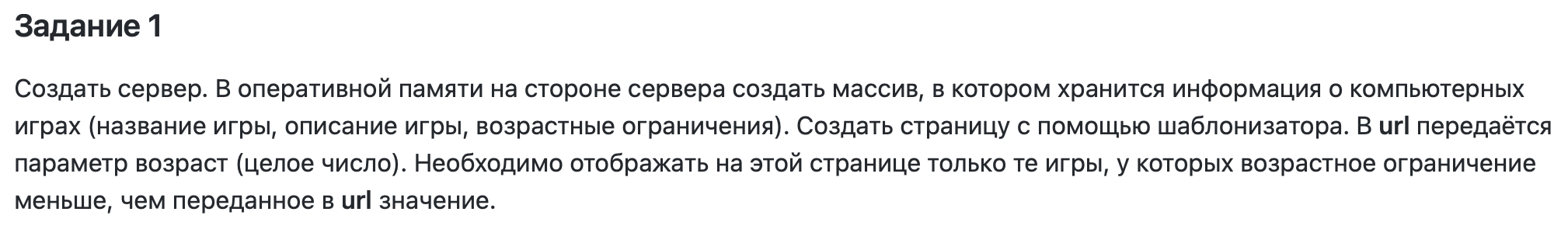
.input-class {

background-color: rgb(30, 30, 30); /\* Цвет фона \*/

color: lightgray; /\* Цвет текста \*/

}

Часть 2



"use strict";

// импорт библиотеки

const express = require("express");

const cookieSession = require("cookie-session");

// запускаем сервер

const app = express();

const port = 5015;

app.listen(port);

console.log(`Server on port ${port}`);

// активируем шаблонизатор

app.set("view engine", "hbs");

const way = \_\_dirname + "/static";

app.use(express.static(way));

// заголовки в ответ клиенту

app.use(function(req, res, next) {

res.header("Cache-Control", "no-cache, no-store, must-revalidate");

res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");

res.header("Access-Control-Allow-Origin", "\*");

next();

});

const gamesArray = [

{

name: "Blacksad: Under the Skin",

description: "Интерактивный квест по мотивам популярного комикса BlackSad, где главные герои — антропоморфные (похожие на людей) животные. История разворачивается в 50-х годах ХХ века и рассказывает о детективе-коте Джоне Блэксэде. Он вместе с дочерью расследует групповое убийство. Жертвы — владелец боксерского клуба и его помощник.",

requiredAge: 18

},

{

name: "Mechwarrior 5: Mercenaries",

description: "Шутер от первого лица, где геймер может почувствовать себя боевым роботом! События происходят в 3015 году — во времена третьей галактической Войны за наследие. Игроки будут выполнять роль оператора специальных роботов. Но это не бездушная «стрелялка»! По ходу сюжета придется выбирать контракты, зарабатывать деньги, чтобы открывать новые детали, да еще и успевать продвигаться по карьерной лестнице.",

requiredAge: 16

},

{

name: "Red Dead Redemption 2",

description: "Эта игра успела менее чем за год перейти в статус «культовой», потому что она смешала в себе культуру дикого запада и дерзость Rockstar.\nКак и в GTA V, здесь есть как одиночный сюжетный, так и многопользовательский онлайн-режим, что позволяет «выходить на охоту» за плохишами, убегать от федералов с головорезами и грабить богачей целыми бандами. Также авторы игры не забыли про графику: увидев степень проработки каждой мелочи, геймерам стало понятно, почему разработка затянулась на целых 5 лет.",

requiredAge: 18

},

{

name: "Need For Speed Heat",

description: "Продолжение гоночной саги: в этот раз — в жарком Майами. Геймеров ждут сумасшедшие гонки на дорогах, побеги от полиции и невероятные вариации для кастомизации авто. Полицейские стали злее — от них некуда спрятаться. Они будут следовать за игроком по пятам. Не поможет даже крушение рекламного щита и заносы на резких поворотах.",

requiredAge: 12

},

{

name: "Call Of Duty: Modern Warfare",

description: "Служба зовет! В 16-ой игре военной франшизы Call of Duty игроки смогут перенестись в Сирию, Ирак и даже на Донбасс. Чтобы избежать моральных проблем, разработчики абстрагировались от реальности, частично изменив события внутри игрового мира.",

requiredAge: 18

},

{

name: "Trine 4: The Nightmare Prince",

description: "Продолжение одного из самых популярных 2.5D-платформеров. Перед игроками окажется знакомая троица: рыцарь Понтий, воровка Зоя и волшебник Амадей. Они будут бороться против монстров, которых случайно выпустил на волю принц Селиус. Теперь троице нужно успеть победить врагов до того времени, как кошмары захватят Вселенную.",

requiredAge: 12

},

{

name: "Fifa 20",

description: "Fifa 20 — продолжение главной футбольной игры с обновленной системой искусственного интеллекта. Матч похож на реальный. В игрушке улучшили работу с мячом и без него. Продумали тщательную и реалистичную физику самого мяча. Также были улучшены высокие пасы, дриблинг, попытки отдать, перехватить мяч и многое другое. Бежать по полю стало так, что аж дух захватывает.",

requiredAge: 0

},

{

name: "Borderlands 3",

description: "Триквел самого безумного шутера последнего десятилетия рассказывает о четверке Искателей Убежища, противостоящих злодеям-близнецам. ",

requiredAge: 18

},

{

name: "The Blackout Club",

description: "Кооперативный хоррор про тайны странного городка, расположенного в секретной военной зоне. И компанию подростков, которая решила стоять на страже малой Родины. По сюжету, геймерам придется объединиться в команду до 4 человек, чтобы защитить город от странного коллективного лунатизма. ",

requiredAge: 16

}

]

app.get("/page/games", function(request, response) {

let age = request.query.age;

if (!age) {

age = "0";

}

const ageInt = parseInt(age);

const resGamesArray = [];

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

if (ageInt >= gamesArray[i]["requiredAge"]) {

resGamesArray.push(gamesArray[i]);

}

}

const infoObject = {

title: "Список игр",

gamesArray: resGamesArray

};

response.render("task1.hbs", infoObject);

});

// работа с сессией

app.use(cookieSession({

name: 'session',

keys: ['hhh', 'qqq', 'vvv'],

maxAge: 24 \* 60 \* 60 \* 1000 \* 365

}));

function findUserIndex(login) {

let reqIndex = -1;

for (let i = 0; i < usersArray.length && reqIndex === -1; i++) {

if (usersArray[i]["login"] === login) {

reqIndex = i;

}

}

return reqIndex;

}

// сохранить cookie

app.get("/api/save", function(request, response) {

// получаем параметры запроса

const login = request.query.login;

const password = request.query.password;

let res = ""

// контролируем существование параметров

if(!login) return response.end(JSON.stringify({result: "Login not set"}));

if(!password) return response.end(JSON.stringify({result: "Password not set"}));

// выставляем cookie

let index = findUserIndex(login)

if (index === -1) return response.end(JSON.stringify({result: "User not found"}));

if (usersArray[index]["password"] !== password)

return response.end(JSON.stringify({result: "Incorrect password"}));

request.session.login = login;

request.session.password = password;

// отправляем ответ об успехе операции

response.end(JSON.stringify({result: "Success"}));

});

// получить cookie

app.get("/api/get", function(request, response) {

// контролируем существование cookie

if(!request.session.login) return response.end(JSON.stringify({result: "You are not authorised"}));

if(!request.session.password) return response.end(JSON.stringify({result: "You are not authorised"}));

// отправляем ответ с содержимым cookie

const login = request.session.login;

const password = request.session.password;

let index = findUserIndex(login)

const hobby = usersArray[index]["hobby"];

const age = usersArray[index]["age"];

response.end(null);

});

// удалить все cookie

app.get("/api/delete", function(request, response) {

request.session = null;

response.end("OK");

});

app.get("/profile", function(request, response) {

const login = request.session.login;

const password = request.session.password;

let index = findUserIndex(login)

response.render("task2.hbs", usersArray[index]);

});

const usersArray = [

{

login: "user1",

password: "passWord1",

hobby: "games",

age: 10

},

{

login: "user2",

password: "password2",

hobby: "sport",

age: 20

},

{

login: "user3",

password: "passWord3",

hobby: "chess",

age: 15

},

{

login: "admin",

password: "AdMiN",

hobby: "administrating",

age: 19

},

{

login: "user4",

password: "password4",

hobby: "web-surfing",

age: 26

},

{

login: "user5",

password: "passWord5",

hobby: "sitting home",

age: 11

}

]



