

# Отчет по разделу #5

## Цель работы

Ознакомиться с разработкой RESTful сервиса с использованием AJAX-запросов.

## Задания 1 - 3

### Условие

- Создать сервер. Сервер должен выдавать страницу с тремя текстовыми полями и кнопкой. В поля ввода вбивается информация о почте, фамилии и номере телефона человека. При нажатии на кнопку "Отправить" введенная информация должна отправляться с помощью POST запроса на сервер и добавляться к концу файла (в файле накапливается информация). При этом на стороне сервера должна происходить проверка: являются ли почта и телефон уникальными. Если они уникальны, то идёт добавление информации в файл. В противном случае добавление не происходит. При отправке ответа с сервера клиенту должно приходить сообщение с информацией о результате добавления (добавилось или не добавилось). Результат операции должен отображаться на странице.
- Добавить серверу возможность отправлять клиенту ещё одну страницу. На данной странице должно быть поле ввода и кнопка. В поле ввода вводится почта человека. При нажатии на кнопку "Отправить" на сервер отправляется GET запрос. Сервер в ответ на GET запрос должен отправить информацию о человеке с данной почтой в формате JSON или сообщение об отсутствии человека с данной почтой.
- Оформить внешний вид созданных страниц с помощью CSS. Информация со стилями CSS для каждой страницы должна храниться в отдельном файле. Стили CSS должны быть подключены к страницам.

### Код программы

Язык: **Javascript**

src/app.js

```
const express = require("express");
const path = require("path");
const body_parser = require("body-parser");

const controllers = require("../controllers");
const router = require("../router");

const PAGES_FOLDER = "public";

const port = process.env.PORT | 3000;

const app = express();

// заголовки в ответ клиенту
app.use(controllers.set_headers);

app.use(body_parser.json());
//app.use(body_parser.urlencoded({ extended: false }))
app.use(controllers.logger);
app.use(express.static(path.join(__dirname, "..", PAGES_FOLDER)));
app.use(body_parser.urlencoded({ extended: true }));

app.use(router);

app.listen(port);
console.log(`Running on port ${port}`);
```

src/controllers.js

```
"use strict";

const path = require("path");
const utils = require("../utils");

const MAIN_PAGE = "index.html";
const SECOND_PAGE = "second.html";
const PAGES_FOLDER = "public";
const PATH_TO_MAIN_PAGE = path.join(__dirname, '..', PAGES_FOLDER, MAIN_PAGE);
```

```

module.exports.default_controller = (_, res) => res.sendFile(PATH_TO_MAIN_PAGE);

module.exports.set_headers = (_, 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();
}

module.exports.add_user = (req, res) => {
  let user = req.body;
  const all_users = get_all();

  const target = all_users.find(value => value.email == user.email && value.phone == user.phone);
  let code = 200;
  let result = "Successfully added!";

  if (!target) {
    utils.append(user);
  } else {
    code = 401;
    result = "Found one!"
    user = target;
  }

  res.status(code).send(JSON.stringify({
    result,
    ...user
  }));
};

const find_user = email => {
  const users = utils.get_all();
  return users.find(user => user.email === email);
}

module.exports.find_user = (req, res) => {
  const target = find_user(req.query.email);
  console.log(req.query);

  if (target) {
    res.status(200).send(JSON.stringify({
      result: "User is found!",
      ...target
    }));
  } else {
    res.status(404).send(JSON.stringify({
      result: "Requested user not found!"
    }));
  }
}

module.exports.logger = (req, _, next) => {
  console.log(req.url, req.method);

  if (req.method === "POST")
    console.log(` body:\n${JSON.stringify(req.body, null, 4)}`);
  else if (req.method === "GET")
    console.log(` query params:\n${JSON.stringify(req.query, null, 4)}`);

  next();
}

module.exports.second_page = (_, res) => {

```

```
    res.sendFile(path.join(__dirname, "..", PAGES_FOLDER, SECOND_PAGE));  
  }  
}
```

#### src/router.js

```
"use strict";  
  
const express = require("express");  
const controllers = require("../controllers");  
  
const router = express.Router();  
  
router.get("/second-page", controllers.second_page)  
router.post("/user", controllers.add_user);  
router.get("/search", controllers.find_user)  
router.get("/", controllers.default_controller);  
  
module.exports = router;
```

#### src/utlis.js

```
const path = require("path");  
const fs = require("fs");  
  
const DB_FILE = path.join(__dirname, "..", "temp", "db.txt");  
  
get_all = () => {  
  return JSON.parse(fs.readFileSync(DB_FILE));  
}  
  
module.exports.append = (object) => {  
  const objects = get_all();  
  objects.push(object);  
  fs.writeFileSync(DB_FILE, JSON.stringify(objects));  
}  
  
module.exports.get_all = get_all;
```

#### public/index.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lab 05</title>
  <link rel="stylesheet" href="assets/styles/style.css">
  <script defer src="assets/scripts/app.js"></script>
</head>
<body>
  <label for="email"></label>
  <input placeholder="Input email..." id="email-input" type="email" name="email">
  <br>
  <label for="lastname"></label>
  <input placeholder="Input lastname..." id="lastname-input" type="text" name="lastname">
  <br>
  <label for="phone"></label>
  <input placeholder="Input phone number..." id="phone-input" type="text" name="phone">
  <br>
  <button>Add!</button>
  <br>

  <div>
    <p id="result-holder"></p>
  </div>

  <a href="/second-page">Search user</a>

</body>
</html>

```

#### public/second.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lab 05</title>
  <link rel="stylesheet" href="assets/styles/style.css">
  <script defer src="assets/scripts/second.js"></script>
</head>
<body>
  <input type="text" id="email-input"></input>
  <button>Search!</button>
  <div>
    <p id="result-holder"></p>
  </div>

  <a href="/">Add user</a>
</body>
</html>

```

#### public/style.css

```

html {
  margin: 0;
  padding: 0;
}

body {
  padding-top: 2em !important;
  margin: 0;
  padding: 0;
  background-color: #060818;
}

```

```
background-color: #000000;
color: #fa00ff !important;
}

label {
  font-size: 1.5em;
  font-weight: bold;
}

p {
  font-size: 1.5em;
  text-align: center;
  white-space: pre;
}

input {
  justify-self: center;
  background-color: #88ff88;
  color: #060818;
  padding: 0.5em;
  font-size: 1em;
  margin-bottom: 0.3em;
  border-radius: 0.8em;
}

div, input, button {
  display: block;
  margin: 0 auto;
}

div {
  border-radius: 2em;
  background-color: white;
  width: 60%;
  height: 15em;
  padding: 2em;
  border: 3px solid #ff88ff;
}

p {
  color: #060818;
}

button {
  font-size: 1em;
  cursor: pointer;
  margin-top: 1em;
  background: #00ff00;
  border-radius: 2em;
  padding: 0.5em 1.5em;
  margin-bottom: 3em;
}

h1 {
  margin-top: 2em;
  margin-bottom: 1em;
  width: 70%;
  margin: 0 auto;
  text-align: center;
}

form {
  text-align: center;
  margin-top: 1em;
}
```

```
a {  
  display: block;  
  width: 200px;  
  text-align: center;  
  text-decoration: none;  
  color: #060818;  
  font-size: 1.2em;  
  border-radius: 1em;  
  margin: 0 auto;  
  margin-top: 2em;  
  padding: 0.4em;  
  background-color: #33ff33;  
}
```

scripts/app.js

```

"use strict";

const post = async (url, body) => {
  //return new Promise((resolve, _) => {
    //let r = new XMLHttpRequest();
    //r.open("POST", url, true);
    //r.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
    //r.send(body);

    //r.onload = function() {
      //resolve(JSON.parse(r.response));
    //}
  //});

  return fetch(url, {
    method: "POST",
    headers: {
      "Content-Type": "application/json;charset=utf-8"
    },
    body
  })
    .then(response => response.json())
  }

const convert_result = (result) => {
  let content = "";
  for (const field in result) {
    content += `${field}: ${result[field]}\r\n`;
  }

  return content;
}

const send_data = () => {
  let email = document.querySelector("#email-input");
  let lastname = document.querySelector("#lastname-input");
  let phone = document.querySelector("#phone-input");
  const label = document.querySelector("#result-holder");

  if (email && lastname && phone && label) {
    email = email.value;
    lastname = lastname.value;
    phone = phone.value;

    const body_string = JSON.stringify({ email, lastname, phone });
    post("/user", body_string)
      .then(result => {
        label.textContent = convert_result(result);
      });

  } else {
    alert("Can't read input");
  }
}

const send_button = document.querySelector("button");
send_button.addEventListener('click', send_data);

```

```

"use strict";

const get = async (url, params) => {
  return new Promise((resolve, _) => {
    url += `?`;

    for (const param in params) {
      url += `${param}=${encodeURIComponent(params[param])}&`
    }

    let r = new XMLHttpRequest();
    r.open("GET", url, true);
    r.setRequestHeader("Content-Type", "text/plain;charset=UTF-8");
    r.send(null);

    r.onload = function() {
      resolve(r.response);
    }
  });
}

const get_by_email = (url, email) => get(url, { email });

const convert_result = (result) => {
  let content = "";
  for (const field in result) {
    content += `${field}: ${result[field]}\r\n`;
  }

  return content;
}

const get_data = () => {
  let email = document.querySelector("#email-input");
  const label = document.querySelector("#result-holder");

  if (email && label) {
    email = email.value;

    get_by_email("/search", email)
      .then(result => {
        result = JSON.parse(result);
        label.textContent = convert_result(result);
      });

  } else {
    alert("Can't read input");
  }
}

const send_button = document.querySelector("button");
send_button.addEventListener('click', get_data);

```

## Результаты тестирования

Все тесты пройдены успешно.

## Вывод



В результате работы был разработан **RESTful** сервис с применением **AJAX**-запросов, выяснены плюсы и минусы использования асинхронных запросов.

# Отчет по разделу #6

## Цель работы

Ознакомиться с работой шаблонизаторов и управлением сессиями в **Node.js**.

## Задания 1-2

- Создать сервер. В оперативной памяти на стороне сервера создать массив, в котором хранится информация о компьютерных играх (название игры, описание игры, возрастные ограничения). Создать страницу с помощью шаблонизатора. В url передаётся параметр возраст (целое число). Необходимо отображать на этой странице только те игры, у которых возрастное ограничение меньше, чем переданное в url значение.
- Создать сервер. В оперативной памяти на стороне сервера создать массив, в котором хранится информация о пользователях (логин, пароль, хобби, возраст). На основе cookie реализовать авторизацию пользователей. Реализовать возможность для авторизованного пользователя просматривать информацию о себе.

## Код программы

Язык: **Javascript**

src/app.js

```
"use strict";

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

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

const controllers = require("./controllers");
const router = require("./router");

const PAGES_FOLDER = path.join(__dirname, "..", "public");
const VIEWS_FOLDER = path.join(__dirname, "..", "views");
const MAX_AGE = 60 * 1000;

const port = process.env.PORT | 3000;

const app = express();

app.set("view engine", "ejs");
app.set("views", VIEWS_FOLDER)

app.use(cookieSession({
  name: 'session',
  keys: ['a', 'b', 'c'],
  maxAge: MAX_AGE
}));

app.use(controllers.set_headers);

app.use(controllers.logger);
app.use(express.static(PAGES_FOLDER));
app.use(body_parser.urlencoded({ extended: true }));

app.use(router);
app.listen(port);

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

src/router.js

```

"use strict";

const express = require("express");
const controllers = require("./controllers");

const router = express.Router();

router.get("/games", controllers.render_games)
router.get("/personal", controllers.render_personal);
router.post("/login", controllers.login);
router.get("/exit", controllers.exit);

router.get("/", controllers.default_controller);

module.exports = router;

```

#### src/controllers.js

```

"use strict";

const PATH_TO_MAIN_PAGE = "index.ejs";
const GAMES_FILE = "games.ejs";
const PERSONAL_FILE = "personal.ejs";

const path = require("path");
const db_obj = require("./db_obj");

module.exports.default_controller = (_, res) => res.render(PATH_TO_MAIN_PAGE);

module.exports.set_headers = (_, 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();
}

module.exports.render_games = (req, res) => {
  let age, games = [], error = null;

  try {
    age = parseInt(req.query.age);
    games = db_obj.games.filter(game => game.age <= age);
  } catch(_) {
    console.log(err);
  }

  if (games.length == 0) {
    error = "There is no games for your query!";
  }

  res.render(GAMES_FILE, { games, error });
}

module.exports.logger = (req, _, next) => {
  console.log(req.url, req.method);

  if (req.method === "POST")
    console.log(` body:\n${JSON.stringify(req.body, null, 4)}`);
  else if (req.method === "GET")
    console.log(` query params:\n${JSON.stringify(req.query, null, 4)}`);

  next();
}

const render_personal = (req, res) => {

```

```

let metainfo = { isAuth: false };

const login = req.session.login;
const password = req.session.password;

if (login && password) {
  metainfo.isAuth = true;
  metainfo.user = db_obj.users.find(elem => elem.login === login && elem.password === password);
}

if (req.error)
  metainfo.error = req.error;

if (!metainfo.error)
  metainfo.error = false;

res.render(PERSONAL_FILE, metainfo);
}

module.exports.login = (req, res) => {
  const login = req.body.login;
  const password = req.body.password;

  if (!login && !password)
    req.error = "No login or password";
  else {
    if (db_obj.users.find(user => user.login === login && user.password === password)) {
      req.session.login = req.body.login;
      req.session.password = req.body.password;
    } else {
      req.error = "No such user in database";
    }
  }

  render_personal(req, res);
}

module.exports.exit = (req, res) => {
  req.session = null;
  res.redirect("/personal");
}

module.exports.render_personal = render_personal;

```

#### src/db\_obj.js

```

"use strict";

module.exports.games = [
  {
    name: "fifa 0+",
    description: "fifa for kids",
    age: 0
  },
  {
    name: "fifa 6+",
    description: "fifa for elder kids",
    age: 6
  },
  {
    name: "fifa 10+",
    description: "fifa for middle preteens",
    age: 10
  },
  {

```

```

    name: "fifa 12+",
    description: "fifa for teens",
    age: 12
  },
  {
    name: "fifa 14+",
    description: "fifa for elder teens",
    age: 14
  },
  {
    name: "fifa 18+",
    description: "fifa for adults",
    age: 18
  },
  {
    name: "fifa 21+",
    description: "fifa for 21+",
    age: 21
  },
  {
    name: "fifa 35+",
    description: "prelast fifa",
    age: 35
  },
  {
    name: "fifa 45+",
    description: "last fifa",
    age: 45
  }
];

```

```

module.exports.users = [
  {
    login: "GULZA",
    password: "gul1",
    hobby: "smurt",
    age: 133
  },
  {
    login: "mrvvz",
    password: "mr1",
    hobby: "Huskell",
    age: 33
  },
  {
    login: "Justarone",
    password: "jus1",
    hobby: "Huckey",
    age: 23
  },
  {
    login: "CATFELLA",
    password: "cat1",
    hobby: "urch",
    age: 18
  },
  {
    login: "hackfeed",
    password: "hac1",
    hobby: "nodejs",
    age: 22
  },
]

```

```
<%- include("includes/header.ejs") %>
  <link rel="stylesheet" href="/style.css">
  <title>Lab 6</title>
</head>
<body>
  <%- include("includes/navbar.ejs", { page: "Home" }) %>
  <div class="simple-card">
    <h1>Games</h1>
    <form method="GET" action="/games">
      <label for="age"></label>
      <input type="text" name="age" placeholder="Enter age...">
      <button type="submit">Check!</button>
    </form>
  </div>
  <div class="simple-card">
    <h1>Personal page</h1>
    <a class="ba" href="/personal">Link to personal pages</a>
  </div>

  <%- include("includes/end.ejs") %>
```

#### views/games.ejs

```
<%- include("includes/header.ejs") %>
  <link rel="stylesheet" href="/style.css">
  <title>Games</title>
</head>
<body>
  <%- include("includes/navbar.ejs", { page: "games" }) %>

  <% if (error) {%>
    <div class="err-card">
      <h1><%=error%></h1>
    </div>
  <% } %>

  <% for (const game of games) { %>
    <div class="simple-card">
      <h1 style="text-align: center;"><%= game.name %></h1>
      <p class="bp">Описание: <%= game.description %></p>
      <p class="bp">Возраст: <%= game.age %></p>
    </div>
  <% } %>

  <%- include("includes/end.ejs") %>
```

#### views/personal.ejs

```

<%- include("includes/header.ejs") %>
  <link rel="stylesheet" href="/style.css">
  <title>Personal</title>
</head>
<body>
  <%- include("includes/navbar.ejs", { page: "personal" }) %>
  <% if (error) {%>
    <div class="err-card">
      <h1><%=error%></h1>
    </div>
  </%>%>

  <% if (isAuth) {%>
    <div class="simple-card">
      <h1>User</h1>
      <p class="bp">Login: <%= user.login %></p>
      <p class="bp">Password: <%= user.password %></p>
      <p class="bp">Hobby: <%= user.hobby %></p>
      <p class="bp">Age: <%= user.age %></p>
    </div>

    <a class="ba exit-a" href="/exit">Exit</a>

  </%> else {%>

    <div class="simple-card">
      <form METHOD="POST" action="/login">
        <label for="login"></label>
        <input type="text" name="login" spellcheck="false" autocomplete="off">
        <br>

        <label for="password"></label>
        <input type="password" name="password" spellcheck="false" autocomplete="off">
        <br>

        <button type="submit">Login</button>
      </form>
    </div>

  </%>%>

<%- include("includes/end.ejs") %>

```

#### views/include/header.ejs

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

#### views/include/navbar.ejs

```

<header>
  <nav>
    <a class=<%= page === "Home" ? "active" : "passive"%> href="/">Home</a>
    <a class=<%= page === "games" ? "active" : "passive"%> href="/games?page=0">Games</a>
    <a class=<%= page === "personal" ? "active" : "passive"%> href="/personal">Personal pages</a>
  </nav>
</header>

```

#### views/include/end.ejs

```
</body>
</html>
```

## public/style.css

```
html {
  padding: 0;
}

body {
  margin: 0;
  padding: 0;
  background-color: #333333;
}

header {
  background-color: #111333;
  border-radius: 0 0 1em 1em;
  border: 2px solid #FFFFFF;
  border-top: None;
  color: white;
  padding: 1% 10%;
}

nav {
  display: flex;
  flex-direction: space-between;
  text-align: center;
}

nav > a {
  flex: 1;
  display: block;
  color: white;
  text-decoration: None;
}

nav > a:hover {
  text-decoration: underline;
}

input {
  padding: 10px;
  font-size: 20px;
  border-radius: 5px;
  border: 1px solid #FFFFFF;
  background-color: #111333;
  color: white;
}

button[type="submit"] {
  padding: 10px;
  font-size: 20px;
  border-radius: 5px;
  border: 1px solid #FFFFFF;
  background-color: #111333;
  color: white;
  margin-left: 12px;
}

button[type="submit"]:hover {
  border-radius: 5px;
  border: 1px solid #00CC00;
  background-color: #00CC00;
}
```

```
}

::placeholder {
  color: #AAACCC;
}

.simple-card {
  background-color: #111333;
  border-radius: 1em;
  margin: 50px 10%;
  border: 2px solid #FFFFFF;
  color: white;
  padding: 10%;
}

form {
  text-align: center;
}

.simple-card > form > * {
  margin-bottom: 10px;
}

h1 {
  text-align: center;
}

.bp {
  font-size: 25px;
}

.ba {
  display: block;
  text-align: center;
  text-decoration: None;
  color: #AAACCC;
  font-size: 25px;
}

.ba:hover {
  color: #00CC00;
}

.err-card {
  background-color: #AA0000;
  border-radius: 1em;
  margin: 50px 10%;
  border: 2px solid #FFFFFF;
  color: white;
  padding: 1% 10%;
}

.exit-a {
  background-color: #AA0000;
  border-radius: 1em;
  display: inline-block;
  margin: 0 10%;
  border: 2px solid #FFFFFF;
  color: white;
  padding: 1% 10%;
}

.exit-a:hover {
  background-color: #BB0000;
  color: white;
}
```



```
}  
  
.active {  
  color: orange;  
}
```

## Результаты тестирования

Все тесты пройдены успешно.

## Вывод

В результате работы был освоен шаблонизатор **EJS** и работа с сессиями при помощи **express-session**.