**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра Вычислительной техники**

**ОТЧЕТ**

**по лабораторной работе № 7**

**по дисциплине «Web программирование» Тема: «Организация передачи данных между запросами пользователей»**

|  |
| --- |
| Студент гр. 3311 Баймухамедов Р. Р. \_\_\_\_\_\_ |
|  |
| Преподаватель Калмыков М.А. \_\_\_\_\_\_ |

Санкт-Петербург

2025

**Цель работы**

Знакомство с методами передачи информации между соединениями, открываемыми в рамках одного сеанса работы пользователя.

**Выполнение лабораторной работы**

Создадим две страницы

prefs.html

<!doctype html>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1">

<title>Настройки (Cookies)</title>

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

<div class="container">

  <div class="header">

    <h1 style="margin:0;">Ввод имени и цвета (Cookies)</h1>

    <a class="brand" href="/">На главную</a>

  </div>

  <form id="prefs-form" class="form card">

    <div>

      <label for="username">Имя пользователя</label>

      <input id="username" class="input" required placeholder="student">

    </div>

    <div>

      <label for="color">Цвет страницы (hex)</label>

      <input id="color" class="input" placeholder="#e6f7ff">

    </div>

    <div>

      <label><input type="checkbox" id="persist"> Запомнить на 7 дней</label>

    </div>

    <div class="toolbar">

      <button class="btn primary">Сохранить в Cookie</button>

      <a class="btn" href="/view.html">Показать значения</a>

    </div>

    <div id="status" class="status"></div>

  </form>

</div>

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

<script src="/i18n-utils.js"></script>

<script>

*// Инициализация интернационализации*

document.addEventListener('DOMContentLoaded', async () => {

  await initI18n();

  updatePrefsForm();

});

*// Обработчик формы*

document.getElementById('prefs-form').addEventListener('submit', async (*e*) => {

*e*.preventDefault();

  const st = document.getElementById('status');

  st.className='status';

  st.textContent = t('prefs\_saving\_status');

  const payload = {

    username: document.getElementById('username').value.trim(),

    color: document.getElementById('color').value.trim(),

    persistDays: document.getElementById('persist').checked ? 7 : undefined

  };

  try {

    const r = await fetch('/api/prefs', { method:'POST', headers:{'Content-Type':'application/json'}, body: JSON.stringify(payload) });

    const data = await r.json();

    if (!r.ok) throw new Error(data.error || ('HTTP '+r.status));

    st.className='status ok';

    st.textContent = t('prefs\_saved\_status');

  } catch (e2) {

    st.className='status err';

    st.textContent = t('prefs\_error\_prefix') + e2.message;

  }

});

</script>

view.html

<!doctype html>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1">

<title>Просмотр Cookie и Session</title>

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

<div class="container">

  <div class="header">

    <h1 style="margin:0;">Cookie + Session</h1>

    <a class="brand" href="/">На главную</a>

  </div>

  <div id="info" class="card">

    <p>Загружаю…</p>

  </div>

</div>

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

<script src="/i18n-utils.js"></script>

<script>

*// Инициализация интернационализации и загрузка данных*

document.addEventListener('DOMContentLoaded', async () => {

  await initI18n();

  await loadViewData();

});

</script>

server.js

app.use(session({

  name: "sid",

  secret: process.env.SESSION\_SECRET || "dev\_change\_me",

  resave: false,

  saveUninitialized: true,

  cookie: {

    httpOnly: true,

    sameSite: "lax",

    secure: false,         *// мы ниже установим динамически по X-Forwarded-Proto*

    maxAge: 30 \* 60 \* 1000 *// 30 минут простоя, как в методичке по умолчанию у Tomcat*

  }

}));

*// POST /api/prefs — сохранить имя пользователя и цвет в Cookie*

*// body: { username, color, persistDays? }*

*// persistDays: если задано — постоянные cookie, иначе — только на сессию*

app.post(`${API\_PREFIX}/prefs`, (*req*, *res*) => {

  const username = String(*req*.body?.username || "").trim();

  let color = String(*req*.body?.color || "").trim();

  const persistDays = Number.isFinite(+*req*.body?.persistDays) ? +*req*.body.persistDays : null;

  if (!username) return *res*.status(400).json({ error: 'Field "username" is required' });

*// простая валидация цвета*

  const validHex = /^#?[0-9a-f]{6}$/i.test(color);

  if (!validHex) color = "#ffffff";

  if (!color.startsWith("#")) color = "#" + color;

  const isHttps = (*req*.headers["x-forwarded-proto"] || "").toLowerCase() === "https";

  const cookieBase = {

    path: "/",

    sameSite: "lax",

    httpOnly: false, *// чтобы фронт мог прочитать при желании; для безопасности лучше true + читать через /api/prefs*

    secure: isHttps

  };

  const age = persistDays ? persistDays \* 24 \* 60 \* 60 \* 1000 : undefined; *// ms*

  const optUser = { ...cookieBase, ...(age ? { maxAge: age } : {}) };

  const optColor = { ...cookieBase, ...(age ? { maxAge: age } : {}) };

*res*.cookie("username", encodeURIComponent(username), optUser);

*res*.cookie("page.color", encodeURIComponent(color), optColor);

*// заодно установим куку с языком (текущий механизм уже это делает, но пусть будет тут)*

  const lang = pickLang(*req*);

*res*.cookie("lang", lang, { ...cookieBase, maxAge: 365\*24\*60\*60\*1000 });

  return *res*.json({ ok: true, cookies: { username, color, lang } });

});

*// GET /api/prefs — отдать содержимое куки (username, color)*

app.get(`${API\_PREFIX}/prefs`, (*req*, *res*) => {

  const raw = *req*.cookies || {};

  const out = {

    username: raw["username"] ? decodeURIComponent(raw["username"]) : null,

    color: raw["page.color"] ? decodeURIComponent(raw["page.color"]) : null,

    lang: raw["lang"] || null

  };

*res*.json({ prefs: out });

});

*// GET /api/session — отдать переменные сессии (visits, lastVisit, id)*

app.get(`${API\_PREFIX}/session`, (*req*, *res*) => {

*res*.json({

    session: {

      id: *req*.sessionID,

      visits: *req*.session.visits || 0,

      lastVisit: *req*.session.lastVisit || null,

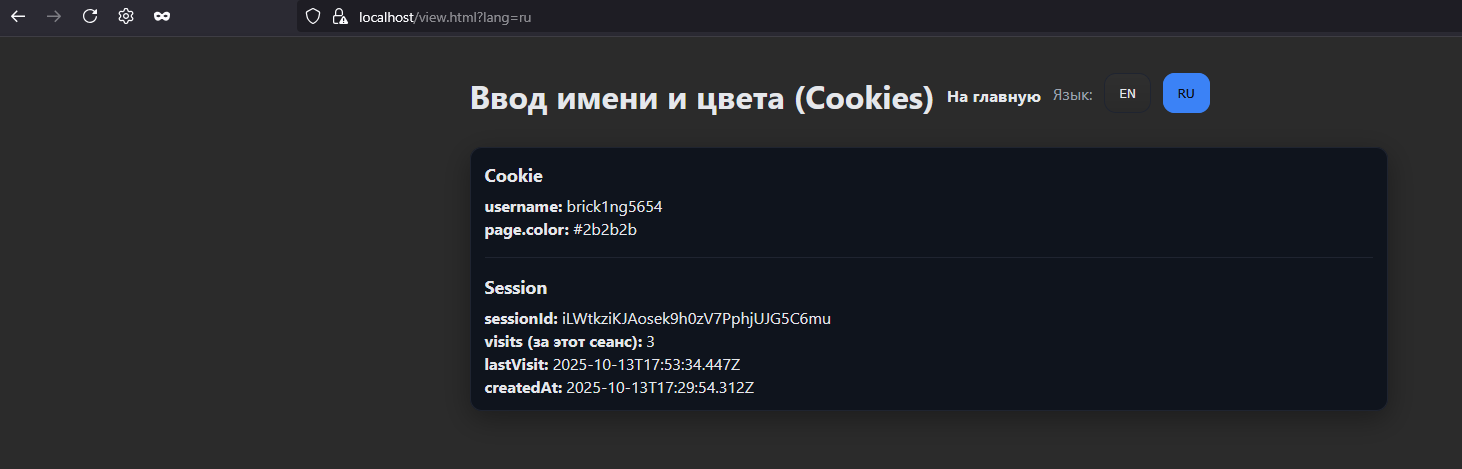
      createdAt: *req*.session.createdAt || null

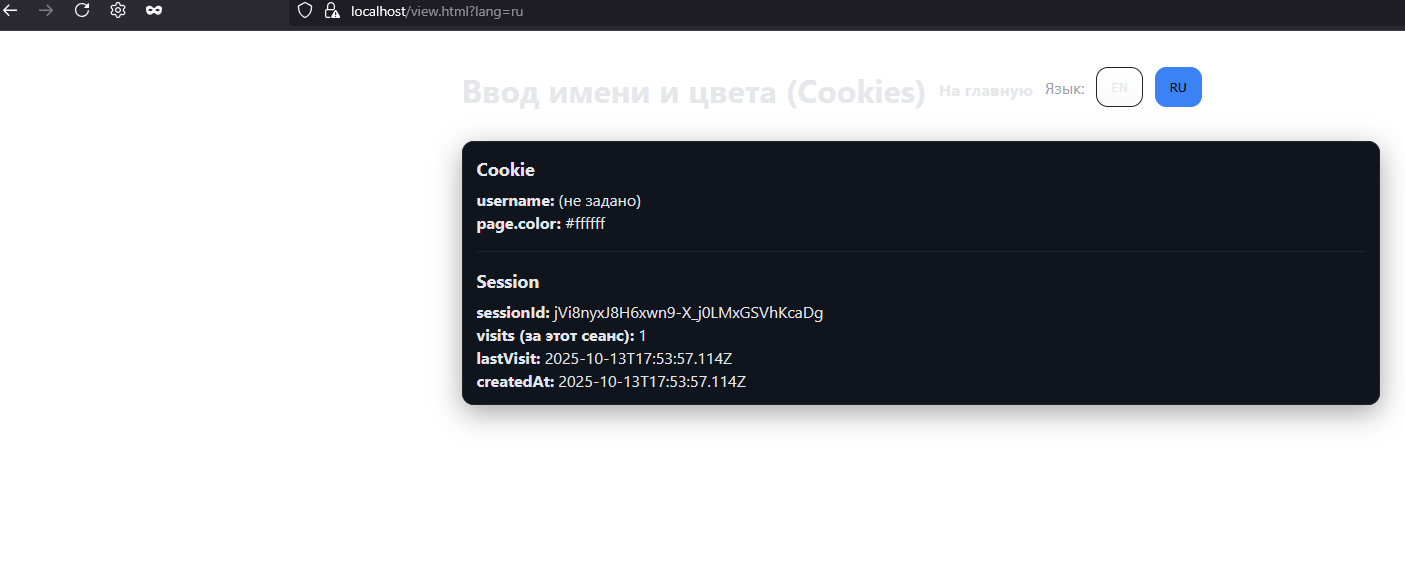
    }

  });

});

**Пример работоспособности программы**





**Заключение**

В ходе выполнения лабораторной работы я ознакомился с методами передачи информации между соединениями, открываемыми в рамках одного сеанса работы пользователя (Cookies).