МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Вычислительной техники

ОТЧЕТ

по лабораторной работе № 5
по дисциплине «Построение Web-приложений с использованием технологий JSP»

Студент гр. 3311 Баймухамедов Р. Р. _____

Преподаватель Калмыков М.А. _____

Санкт-Петербург
2025

Цель работы
Изучить обработку HTTP-запросов GET и POST на Node.js (Express) при разработке простого REST-сервиса.

Выполнение лабораторной работы

Добавим страницы add.html и all.html

add.html

```
<!doctype html>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<title id="doc_title">Добавить фильм</title>
<link rel="stylesheet" href="/style.css" />

<div class="container">
  <div class="header">
    <h1 id="h1_title" style="margin:0;">Добавить фильм (RU + EN)</h1>
    <a class="brand" id="brand_home" href="/">На главную</a>
  </div>

  <div class="toolbar">
    <span class="muted" id="label_lang">Язык:</span>
    <button class="langbtn" data-lang="ru" id="btnRU">RU</button>
    <button class="langbtn" data-lang="en" id="btnEN">EN</button>
    <a class="btn" id="nav_all" href="/all.html" style="margin-left:auto;">Все
фильмы</a>
  </div>

  <form id="add-form" class="form card">
    <div class="row">
      <div>
        <label for="title_ru" id="label_title_ru">Название (RU)</label>
        <input id="title_ru" class="input" required />
      </div>
      <div>
        <label for="title_en" id="label_title_en">Название (EN)</label>
        <input id="title_en" class="input" required />
      </div>
    </div>

    <div class="row">
      <div>
        <label for="year" id="label_year">Год</label>
        <input id="year" type="number" min="1888" max="2100" class="input" />
      </div>
      <div>
```

```
        <label for="rating" id="label_rating">Рейтинг</label>
        <input id="rating" type="number" step="0.1" min="0" max="10"
class="input" />
      </div>
    </div>

    <div>
      <label id="label_genres">Жанры</label>
      <div id="genres_box" class="checklist">
        <label><input type="checkbox" value="action"> <span
id="g_action">Боевик</span></label>
        <label><input type="checkbox" value="scifi"> <span
id="g_scifi">Фантастика</span></label>
        <label><input type="checkbox" value="drama"> <span
id="g_drama">Драма</span></label>
        <label><input type="checkbox" value="comedy"> <span
id="g_comedy">Комедия</span></label>
        <label><input type="checkbox" value="thriller"> <span
id="g_thriller">Триллер</span></label>
        <label><input type="checkbox" value="animation"> <span
id="g_animation">Анимация</span></label>
      </div>
    </div>

    <div>
      <label for="director" id="label_director">Режиссёр</label>
      <input id="director" class="input" placeholder="Лана Вачовски, Лилли
Вачовски" />
    </div>

    <div class="toolbar">
      <button type="submit" class="btn primary" id="btn_save">Сохранить</button>
      <a class="btn" id="link_all" href="/all.html">Все фильмы</a>
    </div>

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

<script>
(function(){
  function getParam(n){return new URLSearchParams(location.search).get(n)||""}
  function setParams(obj, replace=false){
    const url = new URL(location.href);
    Object.keys(obj).forEach(k=>{
      const v=obj[k];
      if(v===null||v===undefined||v==="") url.searchParams.delete(k);
      else url.searchParams.set(k,v);
    });
    history[replace?'replaceState':'pushState'](null,"",url);
  }
```

```
  function getCurrentLang(){
    const fromUrl=getParam('lang'); if(["ru","en"].includes(fromUrl)) return
fromUrl;
    const fromLs=localStorage.getItem('lang'); if(["ru","en"].includes(fromLs))
return fromLs;
    const nav=(navigator.language||'en').slice(0,2); return
["ru","en"].includes(nav)?nav:'en';
  }
  async function loadStrings(){
    const lang=getCurrentLang();
    try{
      window.__i18n = await fetch(`/i18n/${lang}.json?v=6`,{cache:'no-
store'}).then(r=>r.json());
    }catch{
      window.__i18n = (lang==='ru')?{
        brand_home:'На главную', add_title:'Добавить фильм',
add_heading:'Добавить фильм (RU + EN)', label_lang:'Язык:',
        add_label_title_ru:'Название (RU)', add_label_title_en:'Название (EN)',
add_label_year:'Год', add_label_rating:'Рейтинг',
        add_label_genres:'Жанры (через запятую)', add_placeholder_genres:'Боевик,
Фантастика', add_label_director:'Режиссёр', add_placeholder_director:'Лана
Вачовски, Лилли Вачовски',
        add_save:'Сохранить', add_link_all:'Все фильмы',
add_status_saving:'Сохраняю…', add_status_ok:'Ок! Добавлено id=',
add_status_error_prefix:'Ошибка: '
      }:{
        brand_home:'Home', add_title:'Add movie', add_heading:'Add movie (RU +
EN)', label_lang:'Lang:',
        add_label_title_ru:'Title (RU)', add_label_title_en:'Title (EN)',
add_label_year:'Year', add_label_rating:'Rating',
        add_label_genres:'Genres (comma-separated)',
add_placeholder_genres:'Action, Sci-Fi', add_label_director:'Director',
add_placeholder_director:'Lana Wachowski, Lilly Wachowski',
        add_save:'Save', add_link_all:'All movies', add_status_saving:'Saving…',
add_status_ok:'OK! Added id=', add_status_error_prefix:'Error: '
      };
    }
    document.title = window.__i18n.add_title || window.__i18n.app_title ||
document.title;
    const map = [
      ['#doc_title','add_title'], ['#h1_title','add_heading'],
['#brand_home','brand_home'], ['#label_lang','label_lang'],
      ['#nav_all','nav_all'], ['#label_title_ru','add_label_title_ru'],
['#label_title_en','add_label_title_en'],
      ['#label_year','add_label_year'], ['#label_rating','add_label_rating'],
['#label_genres','add_label_genres_simple'],
      ['#label_director','add_label_director'], ['#btn_save','add_save'],
['#link_all','add_link_all']
    ];
    map.forEach(([sel,key])=>{ const el=document.querySelector(sel);
if(el&&window.__i18n[key]!=null) el.textContent=window.__i18n[key]; });
```

```
    // жанры чекбоксы
    const t = window.__i18n || {};
    const text = {
      action: t.genres_action || 'Action',
      scifi: t.genres_scifi || 'Sci-Fi',
      drama: t.genres_drama || 'Drama',
      comedy: t.genres_comedy || 'Comedy',
      thriller: t.genres_thriller || 'Thriller',
      animation: t.genres_animation || 'Animation'
    };
    const ids = {action:'#g_action', scifi:'#g_scifi', drama:'#g_drama',
comedy:'#g_comedy', thriller:'#g_thriller', animation:'#g_animation'};
    Object.keys(ids).forEach(k=>{ const el=document.querySelector(ids[k]); if(el)
el.textContent=text[k]; });
    const director = document.getElementById('director');
if(director&&window.__i18n.add_placeholder_director) director.placeholder =
window.__i18n.add_placeholder_director;
  }

  function setActiveButtons(lang){
    document.getElementById('btnRU').classList.toggle('active', lang==='ru');
    document.getElementById('btnEN').classList.toggle('active', lang==='en');
  }

  document.addEventListener('DOMContentLoaded', async()=>{
    const lang = getCurrentLang();
    if(getParam('lang')!==lang) setParams({lang}, true);
    localStorage.setItem('lang', lang);
    setActiveButtons(lang);
    await loadStrings();

    document.querySelectorAll('.langbtn').forEach(btn=>{
      btn.addEventListener('click', async()=>{
        const newLang = btn.getAttribute('data-lang');
        if(!newLang || newLang===getCurrentLang()) return;
        setParams({lang:newLang}, true);
        localStorage.setItem('lang', newLang);
        setActiveButtons(newLang);
        await loadStrings();
      });
    });

    document.getElementById('add-form').addEventListener('submit', async (e) => {
      e.preventDefault();
      const payload = {
        title_ru: document.getElementById('title_ru').value.trim(),
        title_en: document.getElementById('title_en').value.trim(),
        year: document.getElementById('year').value ?
Number(document.getElementById('year').value) : undefined,
        genres: Array.from(document.querySelectorAll('#genres_box
input[type="checkbox"]:checked')).map(i=>i.value).join(', '),
```

```
        director: document.getElementById('director').value.trim(),
        rating: document.getElementById('rating').value ?
Number(document.getElementById('rating').value) : undefined
      };
      const st = document.getElementById('status');
      st.className = 'status'; st.textContent =
(window.__i18n&&window.__i18n.add_status_saving)||'Saving…';

      try {
        const r = await fetch('/api/movies', {
          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 = ((window.__i18n&&window.__i18n.add_status_ok)||'OK!
Added id=') + data.item_ru.id;
        ['title_ru','title_en','year','director','rating'].forEach(id =>
document.getElementById(id).value = '');
        document.querySelectorAll('#genres_box
input[type="checkbox"]').forEach(i=>i.checked=false);
      } catch (err) {
        st.className = 'status err'; st.textContent =
((window.__i18n&&window.__i18n.add_status_error_prefix)||'Error: ') +
err.message;
      }
    });
  });
})();
</script>
```

all.html

```
<!doctype html>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<title id="doc_title">Все фильмы</title>
<link rel="stylesheet" href="/style.css" />

<div class="container">
  <div class="header">
    <h1 id="h1_title" style="margin:0;">Все фильмы</h1>
    <a class="brand" id="brand_home" href="/">На главную</a>
  </div>

  <div class="toolbar">
    <span class="muted" id="label_lang">Язык:</span>
    <button class="langbtn" data-lang="ru" id="btnRU">RU</button>
```

```
    <button class="langbtn" data-lang="en" id="btnEN">EN</button>
    <a class="btn" id="nav_add" href="/add.html" style="margin-
left:auto;">Добавить фильм</a>
  </div>

  <div id="results" class="grid"></div>
</div>

<script>
(function(){
  const $ = s=>document.querySelector(s);
  const box = document.querySelector('#results');
  function getParam(n){return new URLSearchParams(location.search).get(n)||""}
  function setParams(obj, replace=false){
    const url = new URL(location.href);
    Object.keys(obj).forEach(k=>{
      const v=obj[k];
      if(v===null||v===undefined||v==="") url.searchParams.delete(k);
      else url.searchParams.set(k,v);
    });
    history[replace?'replaceState':'pushState'](null,"",url);
  }
  function getCurrentLang(){
    const fromUrl=getParam('lang'); if(["ru","en"].includes(fromUrl)) return
fromUrl;
    const fromLs=localStorage.getItem('lang'); if(["ru","en"].includes(fromLs))
return fromLs;
    const nav=(navigator.language||'en').slice(0,2); return
["ru","en"].includes(nav)?nav:'en';
  }
  async function loadStrings(){
    const lang=getCurrentLang();
    try{
      window.__i18n = await fetch(`/i18n/${lang}.json?v=6`,{cache:'no-
store'}).then(r=>r.json());
    }catch{
      window.__i18n = (lang==='ru')?{
        brand_home:'На главную', all_title:'Все фильмы', label_lang:'Язык:',
nav_add:'Добавить фильм',
        all_empty:'Пусто', all_loading:'Загружаю…', label_genres:'Жанры:',
label_director:'Режиссёр:', label_rating:'Рейтинг:', label_id:'ID:'
      }:{
        brand_home:'Home', all_title:'All movies', label_lang:'Lang:',
nav_add:'Add movie',
        all_empty:'Empty', all_loading:'Loading…', label_genres:'Genres:',
label_director:'Director:', label_rating:'Rating:', label_id:'ID:'
      };
    }
    document.title = window.__i18n.all_title || window.__i18n.app_title ||
document.title;
```

```
    [["#doc_title","all_title"],["#h1_title","all_title"],["#brand_home","brand_h
ome"],["#label_lang","label_lang"],["#nav_add","nav_add"]]
      .forEach(([sel,key])=>{ const el=document.querySelector(sel);
if(el&&window.__i18n[key]!=null) el.textContent=window.__i18n[key]; });
  }
  function setActiveButtons(lang){
    $("#btnRU").classList.toggle("active", lang==="ru");
    $("#btnEN").classList.toggle("active", lang==="en");
  }
  function card(m){const t=window.__i18n||{};return `
    <div class="card">
      <h3>${m.title} <small class="meta">(${m.year ?? '-'})</small></h3>
      <div class="meta"><b>${t.label_genres||'Genres:'}</b>
${(m.genres||[]).join(', ')}</div>
      <div class="meta"><b>${t.label_director||'Director:'}</b> ${m.director ||
'-'}</div>
      <div class="meta"><b>${t.label_rating||'Rating:'}</b> ${m.rating ?? '-
'}</div>
      <div class="meta"><b>${t.label_id||'ID:'}</b> ${m.id}</div>
    </div>`}
  function render(items){
    if(!Array.isArray(items)||items.length===0){ box.innerHTML=`<p
class="muted">${(window.__i18n&&window.__i18n.all_empty)||'Empty'}</p>`; return;
}
    box.innerHTML = items.map(card).join('');
  }
  async function loadAll(lang, {replace=false}={}){
    await loadStrings();
    setActiveButtons(lang);
    setParams({lang}, replace);
    box.innerHTML = `<div class="card"><span
class="muted">${(window.__i18n&&window.__i18n.all_loading)||'Loading…'}</span></d
iv>`;
    try{
      const r = await fetch(`/api/movies/all?lang=${encodeURIComponent(lang)}`);
      const data = await r.json().catch(()=>({}));
      if(!r.ok) throw new Error(data.error || `HTTP ${r.status}`);
      render(data.items);
    }catch(e){ box.innerHTML = `<div class=\"card status
err\">${e.message}</div>`; }
  }
  document.addEventListener("DOMContentLoaded", ()=>{
    const lang = getCurrentLang();
    if(getParam('lang')!==lang) setParams({lang}, true);
    localStorage.setItem('lang', lang);
    setActiveButtons(lang);
    loadAll(lang, {replace:true});
    document.querySelectorAll(".langbtn").forEach(btn=>{
      btn.addEventListener("click", ()=> {
        const newLang = btn.getAttribute("data-lang");
        if(!newLang || newLang===getCurrentLang()) return;
```

```
        localStorage.setItem('lang', newLang);
        setParams({lang:newLang}, true);
        loadAll(newLang);
      });
    });
  });
  window.addEventListener("popstate", ()=>{
    const l=getParam('lang')||getCurrentLang();
    localStorage.setItem('lang', l);
    loadAll(l, {replace:true});
  });
})();
</script>
```

Добавим переход в index.html

```
<!doctype html>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<title id="doc_title">Movies</title>
<link rel="stylesheet" href="/style.css" />

<div class="container">
<div class="header">
  <h1 id="h1_title" style="margin:0;">Movies</h1>
</div>

<div class="toolbar">
  <span class="muted" id="label_lang">Lang:</span>
  <button type="button" class="langbtn" data-lang="en">EN</button>
  <button type="button" class="langbtn" data-lang="ru">RU</button>

  <span class="spacer"></span>
  <a class="btn" id="nav_add" href="/add.html">Add movie</a>
  <a class="btn" id="nav_all" href="/all.html">All movies</a>
</div>

<form id="search-form" class="form" method="get" action="#">
  <div>
    <label for="q" id="label_search">Search</label>
    <input id="q" class="input" placeholder="" autocomplete="off" />
  </div>
  <div>
    <button class="btn primary" id="search_btn">Search</button>
  </div>
</form>

<div id="results" class="grid" style="margin-top:16px;"></div>
</div>
```

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

Добавим css файл

```
/* ===== Base / Theme ======================================================= */
:root{
  --bg: #ffffff;
  --fg: #0f1115;
  --muted: #6b7280;
  --border: #e5e7eb;
  --accent: #111827;
  --accent-fg: #ffffff;
  --card: #ffffff;
  --ring: 0 0 0 3px rgba(59,130,246,.35);
  --radius: 12px;
  --shadow: 0 6px 24px rgba(0,0,0,.06);
}
@media (prefers-color-scheme: dark){
  :root{
    --bg: #0b0d12;
    --fg: #e5e7eb;
    --muted: #9ca3af;
    --border: #1f2430;
    --accent: #3b82f6;
    --accent-fg: #0b0d12;
    --card: #0f141d;
    --shadow: 0 8px 28px rgba(0,0,0,.35);
  }
}
*{box-sizing:border-box}
html,body{height:100%}
body{
  margin:0; background:var(--bg); color:var(--fg); font:16px/1.45 system-ui, -
apple-system, Segoe UI, Roboto, Inter, Arial, sans-serif;
}

/* ===== Layout ============================================================= */
.container{
  max-width: 950px; margin: 24px auto; padding: 0 16px;
}
.header{
  display:flex; align-items:center; gap:12px; margin: 8px 0 18px;
}
.brand{
  font-weight: 700; letter-spacing:.3px; color:var(--fg); text-decoration:none;
}
.brand:hover{ text-decoration:underline }
```

```
.toolbar{
  display:flex; gap:12px; align-items:center; flex-wrap:wrap;
  margin: 12px 0 16px;
}

.grid{ display:grid; grid-template-columns: 1fr; gap:12px; }
@media (min-width:700px){ .grid{ grid-template-columns: repeat(2, 1fr); }}

/* ===== Buttons / Inputs ============================================== */
.btn, .langbtn{
  display:inline-flex; align-items:center; justify-content:center;
  height:40px; padding:0 14px; border:1px solid var(--border);
  border-radius: var(--radius);
  background: linear-gradient(180deg, rgba(255,255,255,.02), rgba(0,0,0,.02));
  color:var(--fg); cursor:pointer; text-decoration:none; transition:.15s ease;
}
.btn:hover, .langbtn:hover{ transform: translateY(-1px); box-shadow: var(--
shadow); }
.btn:focus-visible, .langbtn:focus-visible{ outline: none; box-shadow: var(--
ring); }
.langbtn.active{
  background: var(--accent); border-color: var(--accent);
  color: var(--accent-fg);
}
.btn.primary{
  background: var(--accent); border-color: var(--accent); color: var(--accent-
fg);
}
.input, input[type="text"], input[type="number"]{
  width:100%; height:40px; padding:0 12px; border:1px solid var(--border);
  border-radius: var(--radius); background:var(--card); color:var(--fg);
}
.input:focus, input[type="text"]:focus, input[type="number"]:focus{
  outline:none; box-shadow: var(--ring); border-color: transparent;
}

/* ===== Cards ========================================================== */
.card{
  border:1px solid var(--border);
  background: var(--card);
  border-radius: var(--radius);
  padding:14px 14px 12px;
  box-shadow: var(--shadow);
}
.card h3{ margin: 0 0 6px; font-size: 18px; }
.meta{ color: var(--muted); font-size: 13px; }

/* ===== Form ============================================================ */
.form{
  display:grid; gap:12px; max-width: 560px;
}
```

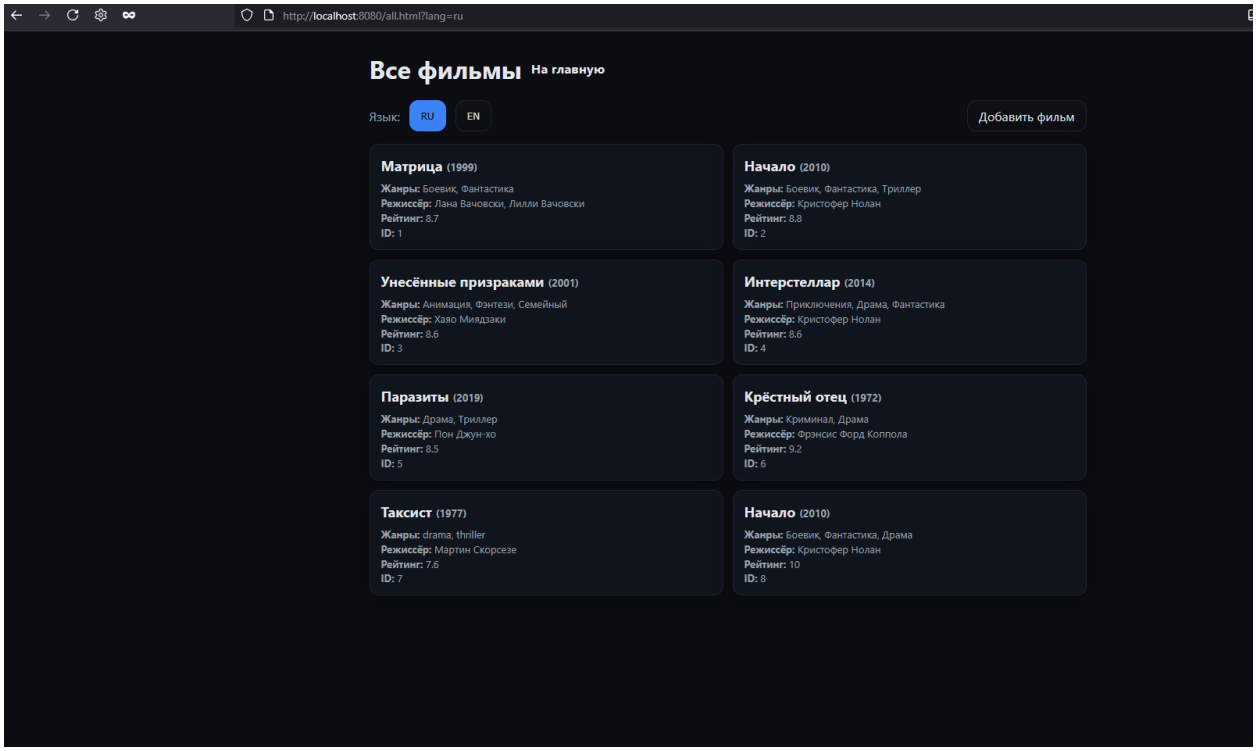
```
.form label{ display:block; font-size: 14px; color: var(--muted); margin-
bottom:6px; }
.form .row{ display:grid; gap:12px; grid-template-columns: 1fr 1fr; }
@media (max-width:640px){ .form .row{ grid-template-columns: 1fr; }}

/* ===== Checklist (genres) ================================================ */
.checklist{
  display:flex; flex-wrap:wrap; gap:8px; margin-top:6px;
}
.checklist > label{
  display:inline-flex; align-items:center; gap:8px;
  padding:8px 12px; border:1px solid var(--border); border-radius:999px;
  background: linear-gradient(180deg, rgba(255,255,255,.02), rgba(0,0,0,.02));
  cursor:pointer; user-select:none; transition:.15s ease;
}
.checklist > label:hover{ transform: translateY(-1px); box-shadow: var(--shadow);
}
.checklist input[type="checkbox"]{ width:16px; height:16px; }
.checklist input[type="checkbox"]:focus-visible{ outline:none; box-shadow: var(--
ring); border-radius:4px; }
.checklist input[type="checkbox"]:checked + span{
  font-weight:600; color: var(--fg);
}

/* ===== Helpers ============================================================ */
.muted{ color: var(--muted); }
.spacer{ height:8px; }
hr{ border:0; border-top:1px solid var(--border); margin:16px 0; }

/* ===== Toast / Status ===================================================== */
.status{ margin-top:8px; font-weight:500; }
.status.ok{ color:#10b981; }
.status.err{ color:#ef4444; }

/* ===== Small badge count ================================================== */
.badge{
  display:inline-flex; align-items:center; justify-content:center;
  min-width:24px; height:24px; padding:0 8px; border-radius:999px;
  background:rgba(59,130,246,.15); color:#60a5fa; font-size:12px; font-
weight:600;
}
```

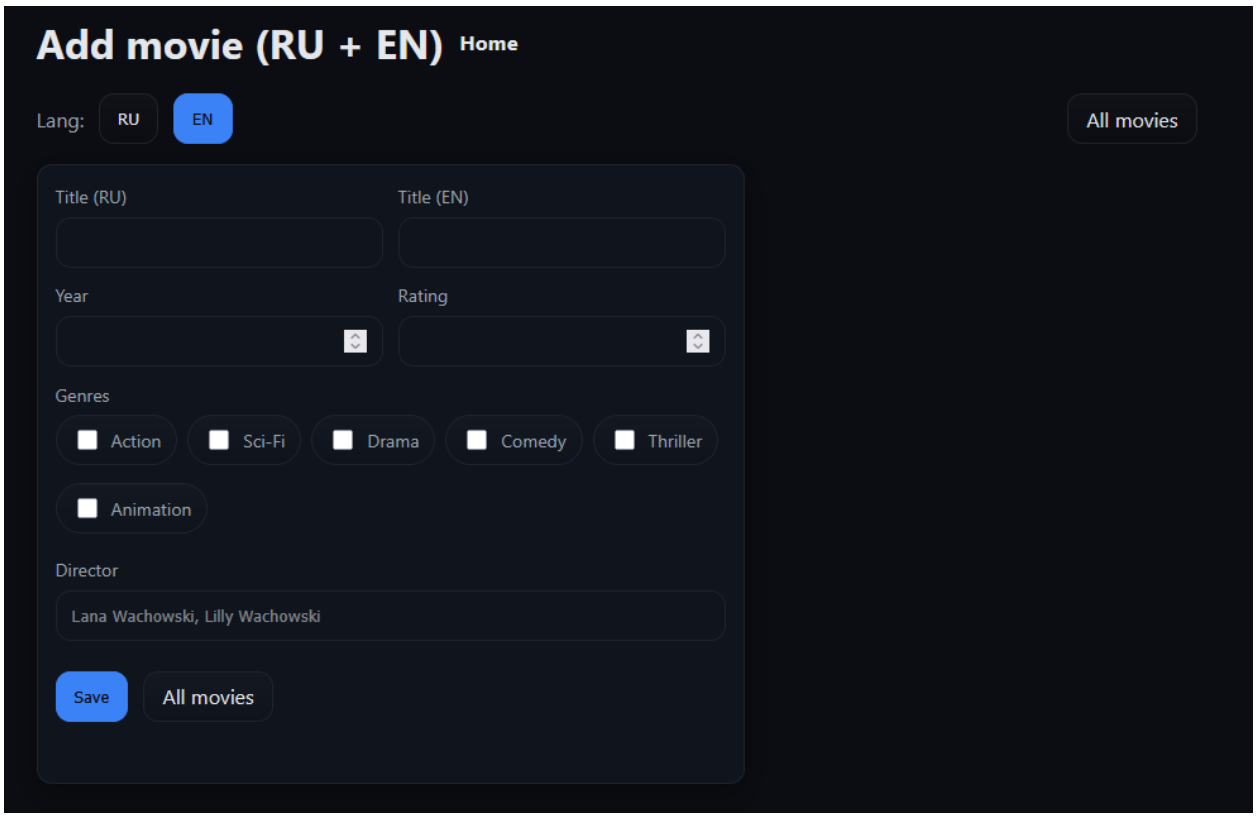
Наиболее важная часть – добавление в server.js эндпоинты GET и POST

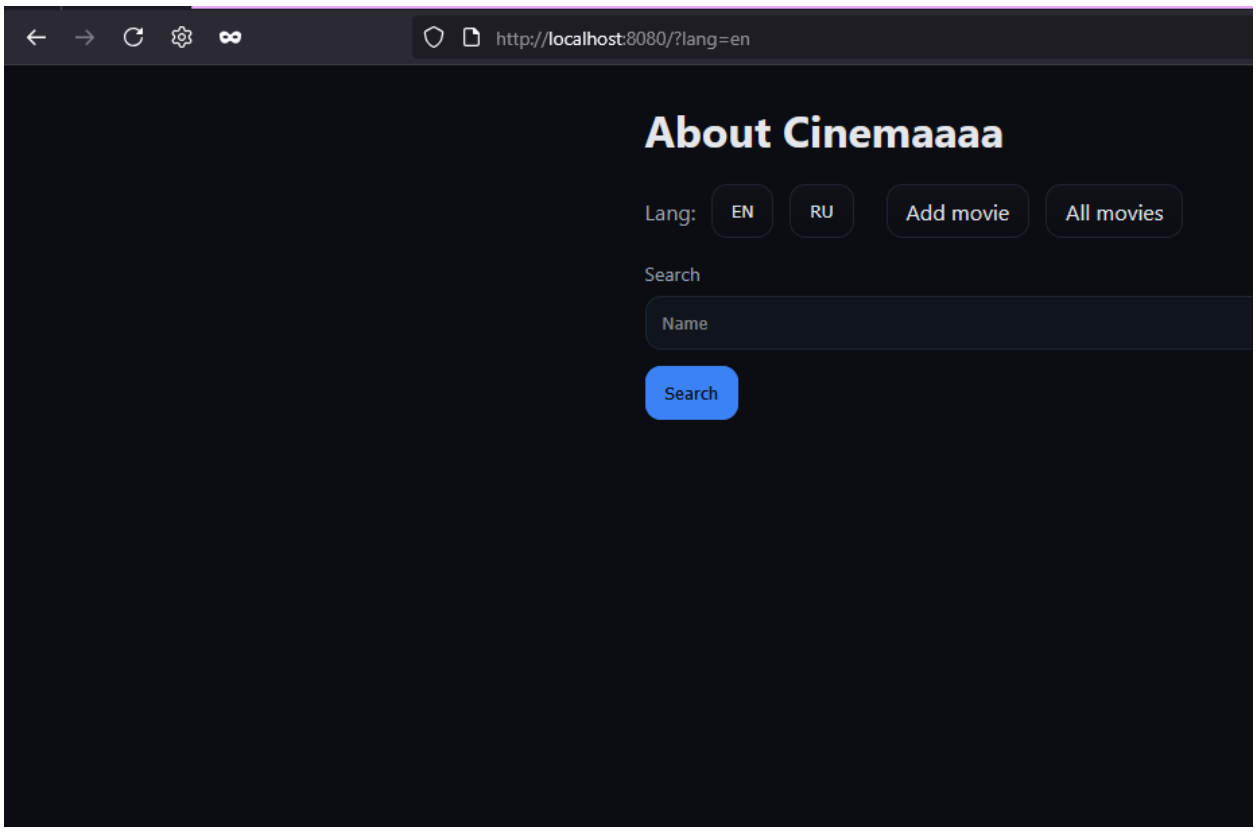
```
// GET /api/movies/all?lang=ru|en  (для all.html)
app.get(`${API_PREFIX}/movies/all`, (req, res) => {
  const lang = pickLang(req);
  const dataset = loadMovies(lang);
```

```
  if (!dataset) return res.status(500).json({ error: "Movies file missing", lang
});
  res.setHeader("Set-Cookie", `lang=${lang}; Path=/; Max-Age=31536000`);
  res.json({ meta: { lang, count: dataset.length }, items: dataset });
});

// POST /api/movies  (добавить запись сразу в ru+en)
app.post(`${API_PREFIX}/movies`, (req, res) => {
  const { ru, en } = loadBoth();
  if (!ru || !en) return res.status(500).json({ error: "Movies files missing" });

  const title_ru = String(req.body?.title_ru || "").trim();
  const title_en = String(req.body?.title_en || "").trim();
  if (!title_ru || !title_en) return res.status(400).json({ error: 'Fields
"title_ru" and "title_en" are required' });

  const year = req.body?.year ? Number(req.body.year) : null;
  // genres могут прийти как строка с кодами ("action, scifi") или массив
  let genres = req.body?.genres;
  if (typeof genres === "string") genres = genres.split(",").map(s =>
s.trim()).filter(Boolean);
  if (!Array.isArray(genres)) genres = [];

  // Преобразуем кодовые значения в подписи для каждой локали
  const ruGenres = genres.map(code => GENRE_LABELS.ru[code] || code);
  const enGenres = genres.map(code => GENRE_LABELS.en[code] || code);
  const director = String(req.body?.director || "").trim() || null;
  const rating = (req.body?.rating !== undefined && req.body.rating !== null) ?
Number(req.body.rating) : null;

  const id = nextIdFromBoth(ru, en);
  const ruMovie = { id, year, genres: ruGenres, director, rating, title: title_ru
};
  const enMovie = { id, year, genres: enGenres, director, rating, title: title_en
};

  ru.push(ruMovie);  en.push(enMovie);
  saveMovies("ru", ru); saveMovies("en", en);

  res.status(201).json({ meta: { created_in: ["ru","en"] }, item_ru: ruMovie,
item_en: enMovie });
});

// ===== 404 ДОЛЖЕН БЫТЬ САМЫМ ПОСЛЕДНИМ ДЛЯ /api =====
app.use(API_PREFIX, (req, res) => res.status(404).json({ error: "Not found" }));

// ===== СТАРТ =====
```

Пример работоспособности программы







Заключение

Изучили обработку HTTP-запросов GET и POST на Node.js (Express) при разработке простого REST-сервиса фильмов