Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Алтайский государственный технический университет им. И.И. Ползунова»

Факультет информационных технологий

Кафедра информационных технологий

Отчёт защищён с оценкой_____

Преподаватель_____ С.В. Умбетов

(подпись) (и.о. фамилия)

« ___»_____2025 г.

Отчёт по лабораторной работе №3

ЛР 12.03.01.15 000 О

Студент группы *1ИСП-22* *Я.А.Далжин*

Преподаватель *к.т.н., доцент* *С.В. Умбетов*

БАРНАУЛ 2025

Задания к лабораторной работе:

1. Написать ретранслятор на node js, сайт с выводом этих данных, данные берутся с json файлов предоставлены dummy.json, при возникновении неправильного адреса , показывать 404.

Задание принял:  *Я.А.Далжин*

Подпись ФИО

1. Написать ретранслятор на node js, сайт с выводом этих данных, данные берутся с json файлов предоставлены dummy. json. при возникновении неправильного адреса , показывать 404.

Далее будут представлен код лабораторной работы, основная часть обработок происходила на стороне клиента в его js файле, ретранслятор считывает url и делает первичную обработку (например, фильтрация во время поиска). Если пользователь вводит неверный url или просто попадает на существующую страницу, то ему высвечивается страница 404.

```
const fs = require("fs");
const http = require("http");
const path = require("path");
const https = require("https");

let products = [];


https.get("https://dummyjson.com/products",  (api) => {
  let data = '';
  api.on('data', chunk => data += chunk);
  api.on('end', () => {
    try {
      const json = JSON.parse(data);
      products = json.products;
    } catch (e) {
      console.error('Ошибка парсинга данных:', e);
    }
  });
}).on('error', (err) => {
  console.error('Ошибка загрузки продуктов:', err);
});

const server = http.createServer((req, res) => {
  const url = req.url;

  if (url.startsWith('/api/')) {
    if (products.length === 0) {
      return res.writeHead(500, { "Content-Type": "text/plain" }).end("Данные ещё не загружены");
    }

    if (url === '/api/products') {
      res.writeHead(200, { "Content-Type": "application/json" });
      return res.end(JSON.stringify(products));
    }


    if (url.startsWith('/api/product/search')) {
      const query = new URLSearchParams(url.split('?')[1]).get('query');
      const filtered = products.filter(p => p.title.toLowerCase().includes(query.toLowerCase()));
      res.writeHead(200, { "Content-Type": "application/json" });
      return res.end(JSON.stringify(filtered));
    }


    const productByIdMatch = url.match(/^\/api\/product\/(\d+)$/);
    if (productByIdMatch) {
      const id = parseInt(productByIdMatch[1], 10);
      const product = products.find(p => p.id === id);
      if (product) {
```

Рисунок 1 – Ретранслятор часть 1

```
const server = http.createServer((req, res) => {

    return res.writeHead(404).end("API не найден");
  }


  let filePath;
  let contentType = "text/html";


  const product_page_match = url.match(/^\/product\/(\d+)$/);
  if (product_page_match) {
    filePath = path.join(__dirname, "ne_temle","page", "product.html");
    contentType = "text/html";
  } else if (url === '/') {
    filePath = path.join(__dirname, "ne_temle", "app.html");
  } else if (url === '/style.css') {
    filePath = path.join(__dirname, "ne_temle", "style.css");
    contentType = "text/css";
  } else if (url === '/404.jpg') {
    filePath = path.join(__dirname, "ne_temle","img","404.jpg");
    contentType = "image/png";
  } else if (url === '/lupa.png') {
    filePath = path.join(__dirname, "ne_temle","img", "lupa.png");
    contentType = "image/png";
  } else if (url === '/script.js') {
    filePath = path.join(__dirname, "ne_temle","script", "scpt.js");
    contentType = "application/javascript";
  } else {
    filePath = path.join(__dirname, "ne_temle","page", "404.html");
  }

  fs.readFile(filePath, (err, content) => {
    if (err) {
      if (err.code === "ENOENT") {
        res.writeHead(404, { "Content-Type": "text/html" });
        filePath = path.join(__dirname, "ne_temle","page", "404.html");
      } else {
        res.writeHead(500);
        res.end("Ошибка сервера");
      }
    } else {
      res.writeHead(200, { "Content-Type": contentType });
      res.end(content);
    }
  });
});

const PORT = 3000;
server.listen(PORT, () => {
  console.log(`Сервер запущен на http://localhost:${PORT}`);
```

Рисунок 2 – Ретранслятор часть 2

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Продукт</title>
    <link rel="stylesheet" href="/style.css">
</head>
<body>
    <div id="content"></div>
    <script src="/script.js"></script>
</body>
</html>
```

Рисунок 3 – Страница одного продукта

```
let allProducts = [];

fetch('/api/products')
  .then(res => res.json())
  .then(products => {
    allProducts = products;
    render_products(products);
  })
  .catch(err => console.error('Ошибка загрузки товаров:', err));


function render_products(products) {
  const container = document.getElementById("content");
  container.innerHTML = '';

  if (products.length === 0) {
    container.innerHTML = '<p>Нет совпадений</p>';
    return;
  }


  update_product_count(products.length);

  products.forEach(product => {
    const card = document.createElement('div');
    card.className = "bord";
    card.innerHTML = `
      <img src="${product.thumbnail}" alt="${product.title}">
      <div class="left">
        <h3 class="name">${product.title}</h3>
        <p class="price">${product.price} $</p>
      </div>
      <button class="right">↗</button>
    `;


    card.addEventListener('click', () => {
      show_product_details(product.id);
    });

    container.appendChild(card);
  });
}


function update_product_count(count) {
  const countElement = document.getElementById("count");
  countElement.textContent = `Товаров - ${count}`;
}
```

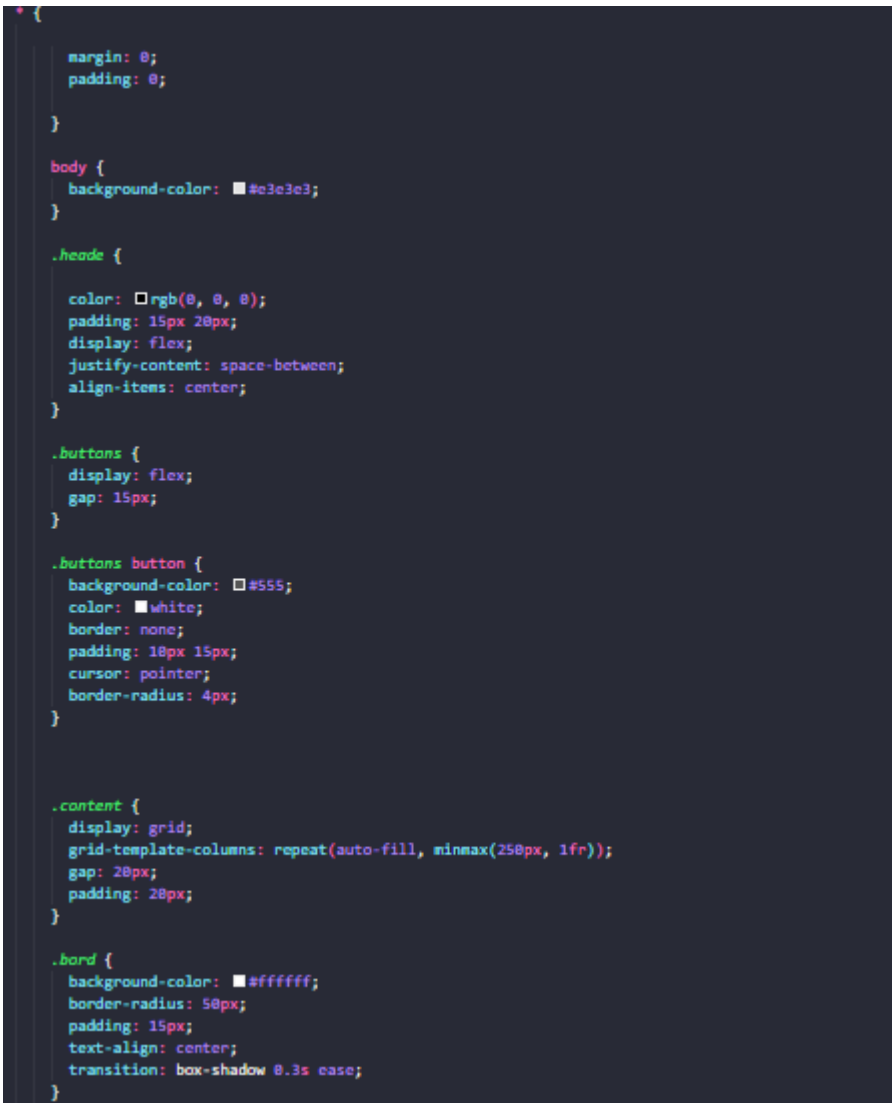
Рисунок 4 – Первая часть скрипта

```
function show_product_details(productId) {
  const product = allProducts.find(p => p.id === productId);

  if (!product) {
    console.error('Товар не найден:', productId);
    return;
  }

  const container = document.getElementById("content");
  container.innerHTML = `
    <div class="prav">
      <div class="single-product">
        <img src="${product.thumbnail}" alt="${product.title}">
        <h2>${product.title}</h2>
        <p>${product.description}</p>
        <p>${product.price} $</p>
      </div>
    </div>
  `;

  history.pushState(null, '', `/product/${productId}`);
}


function poisk() {
  const query = document.getElementById("search").value.toLowerCase();
  if (!query) return;

  const results = allProducts.filter(p => p.title.toLowerCase().includes(query));
  render_products(results);
  history.pushState(null, '', `/search?query=${encodeURIComponent(query)}`);
}
```

Рисунок 5 – Вторая часть скрипта

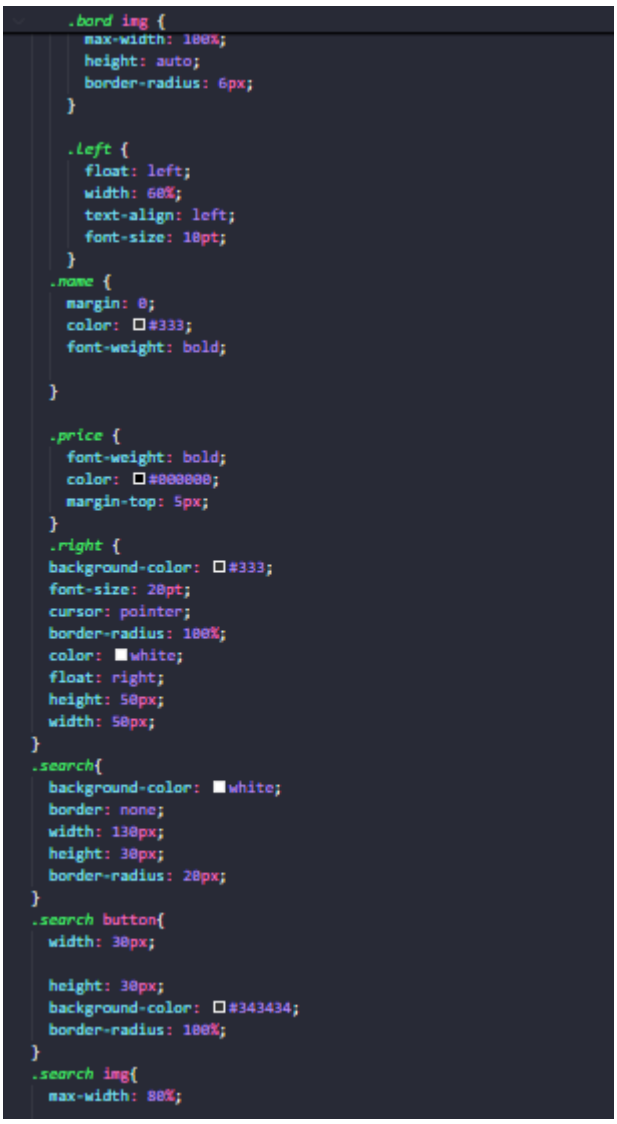
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Товарыс</title>
    <link rel="stylesheet" href="/style.css">
</head>
<body>
    <div class="heade">
        <h1 id="count">Товаров -</h1>
        <div class="search">
            <input type="text" id="search">
            <button onclick="poisk()"><img src="/lupa.png" alt=""></button>
        </div>
    </div>
    <div class="content" id="content"></div>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jszip/3.10.1/jszip.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/FileSaver.js/2.0.5/FileSaver.min.js"></script>
    <script src="/script.js"></script>
</body>
</html>
```

Рисунок 6 – Страница с продуктами



```
* {

    margin: 0;
    padding: 0;

}

body {
    background-color: #e3e3e3;
}

.heade {

    color: rgb(0, 0, 0);
    padding: 15px 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.buttons {
    display: flex;
    gap: 15px;
}

.buttons button {
    background-color: #555;
    color: white;
    border: none;
    padding: 10px 15px;
    cursor: pointer;
    border-radius: 4px;
}



.content {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(250px, 1fr));
    gap: 20px;
    padding: 20px;
}

.bord {
    background-color: #ffffff;
    border-radius: 50px;
    padding: 15px;
    text-align: center;
    transition: box-shadow 0.3s ease;
}
```

Рисунок 7 – Стили первая часть



```
    .bord img {
      max-width: 100%;
      height: auto;
      border-radius: 6px;
    }

    .left {
      float: left;
      width: 60%;
      text-align: left;
      font-size: 10pt;
    }
  .name {
    margin: 0;
    color: #333;
    font-weight: bold;

  }

  .price {
    font-weight: bold;
    color: #800000;
    margin-top: 5px;
  }
  .right {
  background-color: #333;
  font-size: 20pt;
  cursor: pointer;
  border-radius: 100%;
  color: white;
  float: right;
  height: 50px;
  width: 50px;
}
.search{
  background-color: white;
  border: none;
  width: 130px;
  height: 30px;
  border-radius: 20px;
}
.search button{
  width: 30px;

  height: 30px;
  background-color: #343434;
  border-radius: 100%;
}
.search img{
  max-width: 80%;
```

Рисунок 8 – Стили вторая часть

```
  border-radius: 20px;
}
.search button{
  width: 30px;

  height: 30px;
  background-color: #343434
  border-radius: 100%;
}
.search img{
  max-width: 80%;

}
.search input{
  width: 100px;
  height: 100%;
  border: none;
  border-radius: 20px;
  float: left;
}
.single-product {
  background-color: #ffffff
  border-radius: 50px;
  padding: 30px;
  text-align: center;
  box-shadow: 0 4px 20px rg
  max-width: 500px;
  width: 90%;
  margin: 0 auto;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%,
}


.prav {
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  width: 100%;
}
```

Рисунок 9 – Стили 3 часть

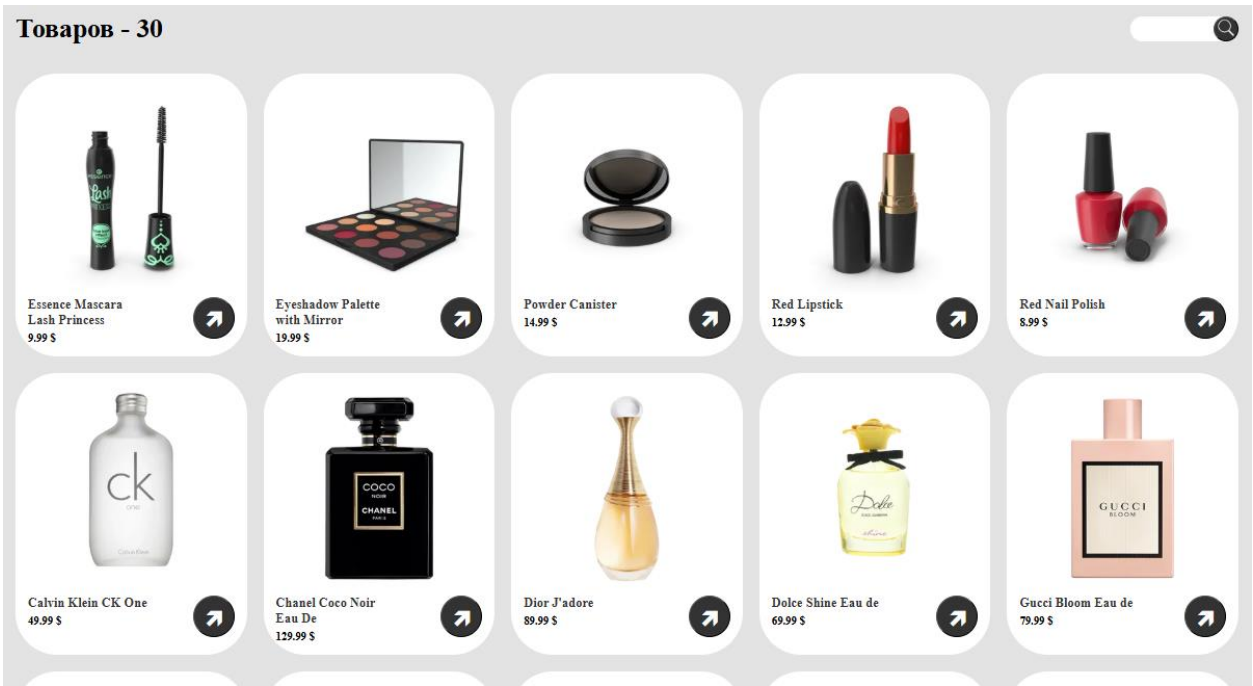


Рисунок 10 – Страница товаров

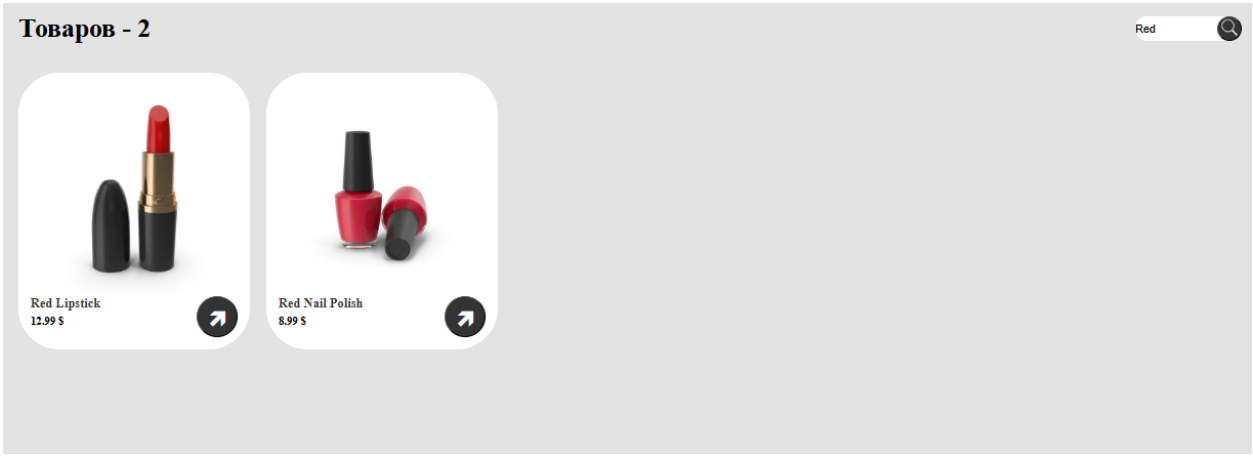


Рисунок 11 – Поиск

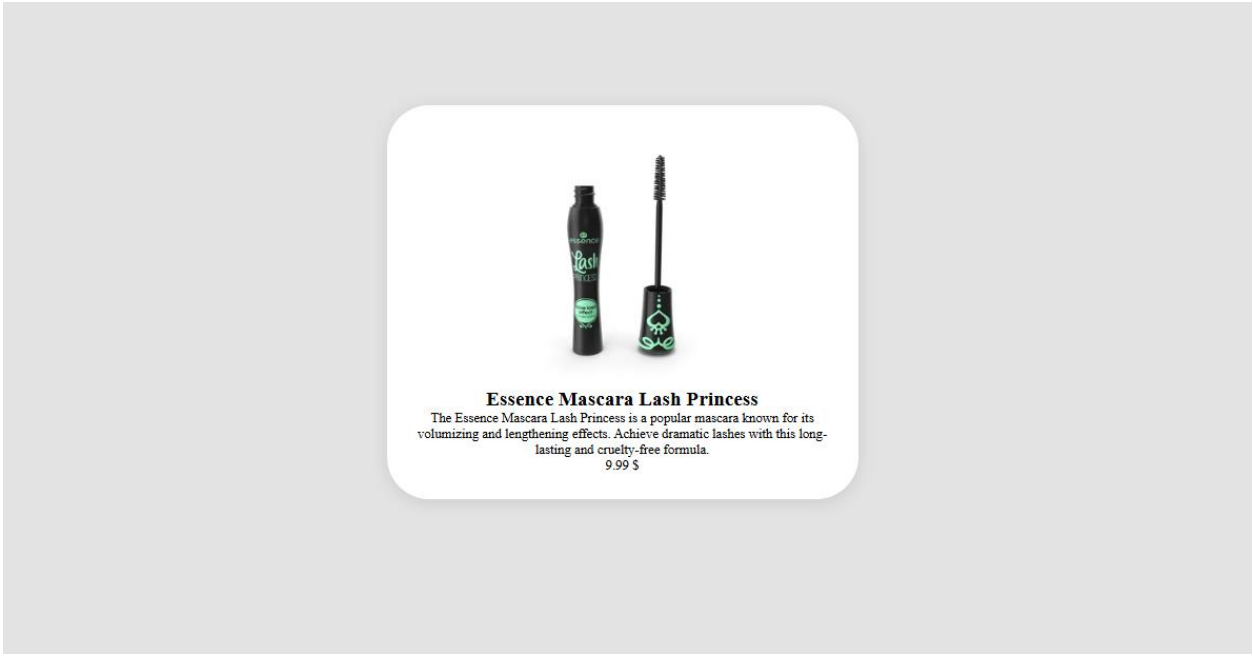


Рисунок 12 – Страница товара

Рисунок 13 – Страница 404

Более подробно можно ознакомиться по ссылке на Github - https://github.com/dalzhin1isp23/-_2025.git

Вывод:

В ходе данной лабораторной работы был создан сайт, на который выводит данные ретранслятор, который так же был написан в ходе лабораторной работы, данные берутся с Dummy. Json - сайт который предоставляет данные для проверки сервера.