



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 09.03.01 Компьютерные системы и сети

## О Т Ч Е Т

по лабораторной работе № 7

Название: Основы Front-End разработки на JavaScript

Дисциплина: Языки Интернет-Программирования

Студент

ИУ6-31Б

(Группа)

\_\_\_\_\_  
(Подпись, дата)

К.Д. Коротаев

(И.О. Фамилия)

Преподаватель

\_\_\_\_\_  
(Подпись, дата)

И.О. Фамилия

(И.О. Фамилия)

Москва, 2024

**Цель работы:** изучение основ разработки SPA-приложение на JavaScript.

**Задание:**

1. Ознакомиться с материалами для подготовки перед выполнением лабораторной работы
2. Сделать форк данного репозитория в GitHub, клонировать получившуюся копию локально, создать от мастера ветку dev и переключиться на неё
3. Реализовать пользовательский веб-интерфейс для взаимодействия с микросервисами, которые были получены в ходе выполнения предыдущей лабораторной работы. Взаимодействие с Back-End частью веб-приложения должно осуществляться с помощью AJAX-запросов.
4. Сделать отчёт и поместить его в директорию docs
5. Зафиксировать изменения, сделать коммит и отправить получившееся состояние ветки dev в личный форк данного репозитория в GitHub
6. Через интерфейс GitHub создать Pull Request dev --> master
7. На защите лабораторной работы продемонстрировать работоспособность приложения через браузер

## Ход работы:

Для выполнения работы были созданы компоненты, отвечающие за создание форм и их связь с Front-end. Каждая форма предназначена для отдельного типа запроса.

### Код для работы с сервисом Hello:

```
import React, { useState } from 'react';

function HelloForm() {
  const [count, setCount] = useState("");

  async function fetchCount(event) {
    event.preventDefault();
    try {
      const response = await fetch('http://localhost:8082/get');
      if (response.ok) {
        const data = await response.text();
        setCount(data);
      } else {
        console.error('Ошибка при получении данных');
      }
    } catch (error) {
      console.error(error);
    }
  }

  return (
    <form>
      <div>
        <button onClick={fetchCount}>Получить привет</button>
        {count} && <p>Привет: {count}</p>
      </div>
    </form>
  );
}

export default HelloForm;
```

### Код для работы с сервисом Query:

```
import React, { useState } from 'react';

function QueryForm() {
  const [name, setName] = useState("");
  const [inputValue, setInputValue] = useState("");

  async function fetchQuery(event) {
    event.preventDefault();
```

```

    try {
      const response = await fetch('http://localhost:8083/api/user?name=' +
inputValue);
      if (response.ok) {
        const data = await response.text();
        setName(data);
      } else {
        console.error('Ошибка при получении данных');
      }
    } catch (error) {
      console.error(error);
    }
  }
}

return (
  <form>
    <input
      type="text"
      value={inputValue}
      onInput={(e) => setInputValue(e.target.value)}
    />
    <button type="submit" onClick={fetchQuery}>Отправить</button>
    {name && <p>Результат: {name}</p>}
  </form>
);
}

export default QueryForm;

```

### **Код для работы с сервисом Count:**

#### ***GetForm.jsx (для GET-запросов) -***

```

import React, { useState } from 'react';

function GetForm() {
  const [count, setCount] = useState("");

  async function fetchCount(event) {
    event.preventDefault();
    try {
      const response = await fetch('http://localhost:8081/count');
      if (response.ok) {
        const data = await response.text();
        setCount(data);
      } else {

```

```

        console.error('Ошибка при получении данных');
    }
} catch (error) {
    console.error(error);
}
}

return (
    <form>
        <div>
            <button onClick={fetchCount}>Получить счетчик</button>
            {count} && <p>Счетчик: {count}</p>
        </div>
    </form>
);
}

```

export default GetForm;

### ***PostForm.jsx (для POST-запросов) -***

```
import React, { useState } from 'react';
```

```
function PostForm() {
    const [value, setValue] = useState();
```

```
    function handleSubmit(e) {
```

```
        const count = parseInt(value);
        if (!isNaN(count)) {
            fetch('http://localhost:8081/count', {
                method: 'POST',
                headers: { 'Content-Type': 'application/x-www-form-urlencoded' },
                body: `count=${value}`,
            }).then(() => {
```

```
                });
            } else {
                alert('Введите число');
            }
            e.preventDefault();
        }
    }

```

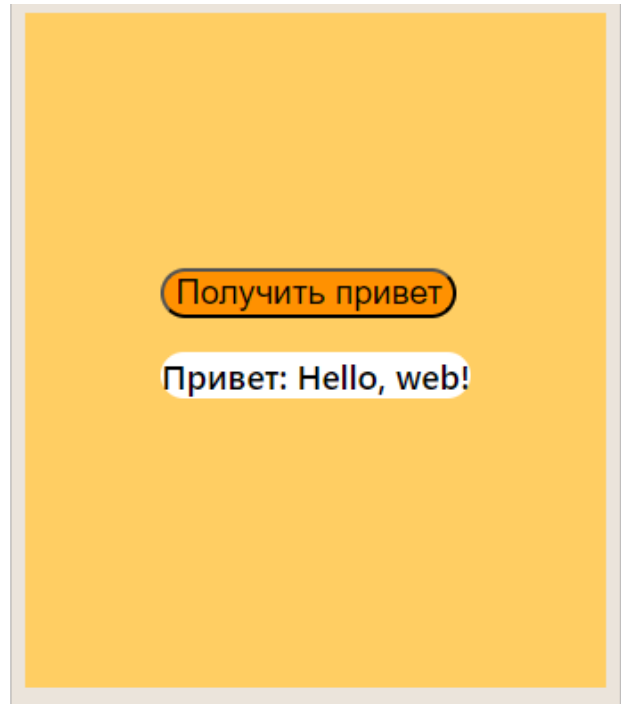
```
    return (
        <form onSubmit={handleSubmit}>
            <input value={value} onChange={(e) => setValue(e.target.value)} />

```

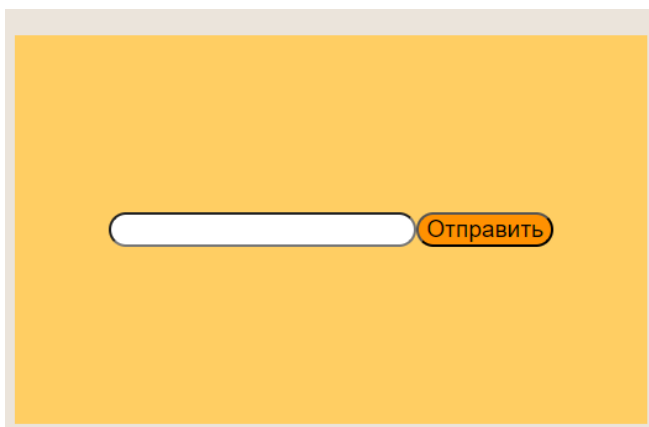
```
        <button type="submit">Добавить значение к счетчику</button>
    </form>
);
}
```

```
export default PostForm;
```

### Тестирование к сервису Hello:



### Тестирование к сервису Query:



## Тестирование к сервису Count:

The image displays two screenshots of a web application interface for testing a counter service. Both screenshots feature a light gray header and a light gray footer. The main content area is divided into two columns by a vertical gray line.

**Top Screenshot:**

- Left Column:** Contains a white input field and a button labeled "Добавить значение к счетчику" (Add value to counter).
- Right Column:** Contains a button labeled "Получить счетчик" (Get counter) and a label "Счетчик: 0" (Counter: 0).

**Bottom Screenshot:**

- Left Column:** The input field now contains the value "5345". The button "Добавить значение к счетчику" remains.
- Right Column:** The button "Получить счетчик" remains, and the label has updated to "Счетчик: 5345" (Counter: 5345).

**Заключение:** для создания интерактивных сайтов удобно использовать библиотеку React для отображения и контроля состояния сайта.