



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 Информатика и вычислительная техника

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)

Д.Ю. Воронин

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

В.Д. Шульман

(И.О. Фамилия)

Москва, 2024

Цель работы – изучение основ разработки SPA-приложения 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 дев 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

Решение

Для решения задачи была разработана функциональная компонента App. Данная компонента позволяет выбрать:

- 1) Программу, к которой будет посылаться запрос
- 2) Соответствующий выбор запросов для выбранной программы
- 3) Параметры запроса, если выбранная программа их предусматривает

В результате компонента выдаёт результат обращения к программе.

Ниже представлено содержание файла App.js

```
import './App.css';
import React from 'react'

const URL = ["localhost:8082", "localhost:8083/api/user?name=",
"localhost:8081/count"]

class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { indexProgram: 0, indexRequest: 0, url: URL[0], inputText: "", result:
"", requestStatus: "" };
  }

  handleChangeProgram(event) {
    this.setState({ indexProgram: event.target.value, url: URL[event.target.value] });
    this.clearFields();
  }

  handleChangeRequest(event) {
    this.setState({ indexRequest: event.target.value });
    this.clearFields();
  }
```

```

clearFields() {
  this.setState({ inputText: "", requestStatus: "", result: "" })
}

handleChangeInput(event) {
  this.setState({ inputText: event.target.value })
  if (this.state.indexProgram == 1) {
    this.setState({ url: URL[1] + encodeURIComponent(event.target.value) })
  }
}

handleClick() {
  this.setState({ requestStatus: "", result: "" })
  if (this.state.indexRequest == 0) {
    fetch(`http://${this.state.url}`).then(async (response) => {
      let responseValue = await response.text();
      let responseStatus = response.status;
      this.setState({ requestStatus: responseStatus, result: responseValue });
    }).catch((err) => {
      this.setState({ requestStatus: 500, result: "Ошибка соединения!" });
    })
  } else {
    let formData = new FormData();
    formData.append('count', this.state.inputText);
    fetch(`http://${this.state.url}`, {
      method: 'POST',
      body: formData
    }).then(async (response) => {
      let responseValue = await response.text();
      let responseStatus = response.status;
      this.setState({ requestStatus: responseStatus, result: responseValue });
    })
  }
}

```

```

    }).catch((err) => {
      this.setState({ requestStatus: 500, result: "Ошибка соединения!" });
    })
  }
}

render() {
  return (
    <div className='App'>
      <div className='head'>
        <div className='head_program'>
          <label className='text'>Программа</label>
          <select className='select' value={this.state.indexProgram}
onChange={(event) => this.handleChangeProgram(event)}>
            <option value="0">Hello</option>
            <option value="1">Query</option>
            <option value="2">Count</option>
          </select>
        </div>
        <button className='button_run' onClick={() =>
this.handleClick()}>Отправить</button>
        <div className='head_request'>
          <label className='text'>Запрос</label>
          <select className='select' value={this.state.indexRequest}
onChange={(event) => this.handleChangeRequest(event)}>
            <option value="0">GET</option>
            {this.state.indexProgram == 2 && <option
value="1">POST</option>}
          </select>
        </div>
      </div>
      <hr className='separator'></hr>
    </div>
  );
}

```

```

    {(this.state.indexProgram == 1 || (this.state.indexProgram == 2 &&
this.state.indexRequest == 1)) &&
    <>
      <div className='body'>
        {this.state.indexProgram == 1 && <>
          <div className='text'><strong>Query Params</strong></div>
          <label className='text'>name: <input className="input"
value={this.state.inputText} onChange={(event) => this.handleChangeInput(event)}
/></label>
          </>
          {this.state.indexProgram == 2 && <>
            <div className='text'><strong>Form Params</strong></div>
            <label className='text'>count: <input className="input"
value={this.state.inputText} onChange={(event) => this.handleChangeInput(event)}
/></label>
            </>
          </div>
          <hr className='separator'></hr>
          </>
        <div className='body'>
          <label className='text'><strong>URL: </strong> {this.state.url}</label>
          <div className='text'><strong>Результат:
</strong>{this.state.result}</div>
          <div className='text'>Статус: {this.state.requestStatus}</div>
        </div>
      </div>
    );
  }
}

```

```
export default App;
```

Тестирование

На рисунках 1-9 представлены результаты тестирования программы.

Программа **Hello** Запрос **GET**

URL: localhost:8082
Результат: Hello, web!
Статус: 200

Рисунок 1 – Результат успешного запроса к программе Hello

Программа **Hello** Запрос **GET**

URL: localhost:8082
Результат: Ошибка соединения!
Статус: 500

Рисунок 2 – Результат отсутствия соединения с программой Hello

Программа **Query** Запрос **GET**

Query Params
name:

URL: localhost:8083/api/user?name=
Результат: Пустая строка!
Статус: 400

Рисунок 3 – Результат неудачного запроса к программе Query

Программа **Query** Запрос **GET**

Query Params
name:

URL: localhost:8083/api/user?name=Java
Результат: Hello, Java!
Статус: 200

Рисунок 4 – Результат успешного запроса к программе Query

Программа **Query** Запрос **GET**

Query Params

name:

URL: localhost:8083/api/user?name=%D0%AF%D0%B2%D0%B0

Результат: Hello, Ява!

Статус: 200

Рисунок 5 – Результат запроса, содержащего буквы кириллицы, к программе Query

Программа **Count** Запрос **POST**

Form Params

count:

URL: localhost:8081/count

Результат: Ошибка соединения!

Статус: 500

Рисунок 6 – Результат отсутствия соединения с программой Count

Программа **Count** Запрос **GET**

URL: localhost:8081/count

Результат: 0

Статус: 200

Рисунок 7 – Результат первого запроса к программе Count

Программа **Count** Запрос **POST**

Form Params

count:

URL: localhost:8081/count

Результат:

Статус: 200

Рисунок 8 – Результат POST запроса к программе Count

The screenshot shows a web application interface with a light blue background. At the top, there is a header bar. On the left, it says "Программа" followed by a dropdown menu showing "Count". In the center, there is a button labeled "Отправить". On the right, it says "Запрос" followed by a dropdown menu showing "GET". Below the header bar, there is a horizontal line. Underneath the line, the following text is displayed: "URL: localhost:8081/count", "Результат: 200", and "Статус: 200".

Рисунок 9 – Результат GET запроса, проверяющего успешность предыдущего POST запроса

Заключение:

Библиотека React позволяет создавать интерактивные сайты, осуществляя при этом взаимодействие с Back-End'ом при помощи AJAX запросов

Список использованных источников:

<https://github.com/ValeryBMSTU/web-7>