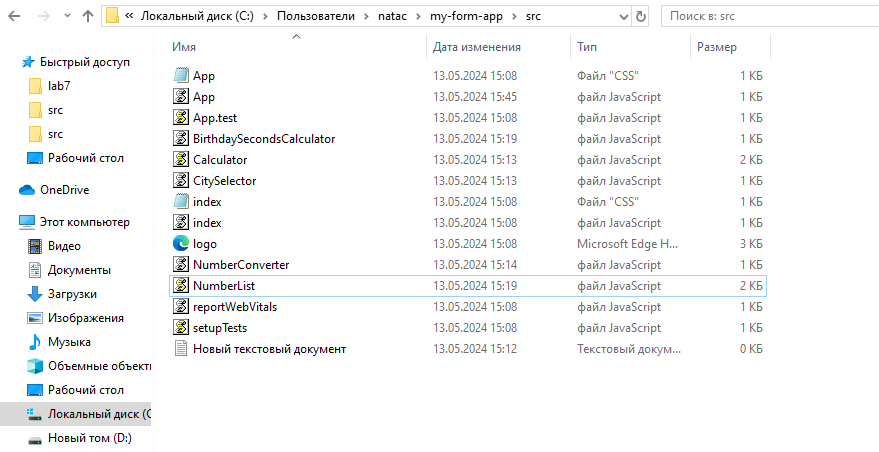
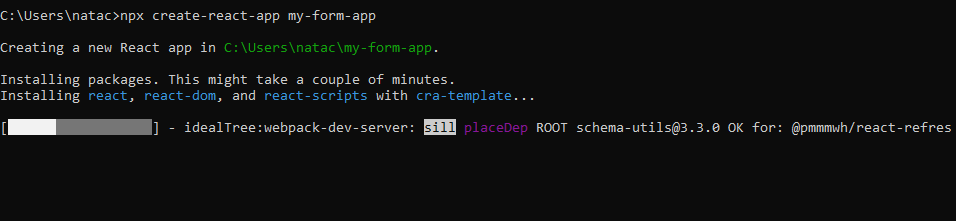
**Лабараторная работа №9**

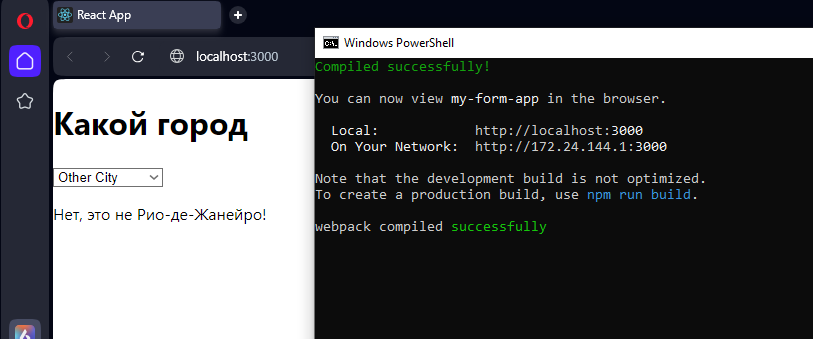
**Создание проекта**



**Задачи про формы**

**1.- Сделать компонент, отображающий селектбокс с городами. Если выбран не Рио, писать снизу: "Нет, это не Рио-де-Жанейро!"**





**App.js**

import React from 'react';

import CitySelector from './CitySelector';

const App = () => {

return (

<div>

<h1>Какой город</h1>

<CitySelector />

</div>

);

};

export default App;

**CitySelector.js**

import React, { useState } from 'react';

const CitySelector = () => {

const [selectedCity, setSelectedCity] = useState('');

const handleCityChange = (e) => {

setSelectedCity(e.target.value);

};

return (

<div>

<select value={selectedCity} onChange={handleCityChange}>

<option value="rio">Rio-de-Janeiro</option>

<option value="other">Other City</option>

</select>

{selectedCity !== 'rio' && <p>Нет, это не Рио-де-Жанейро!</p>}

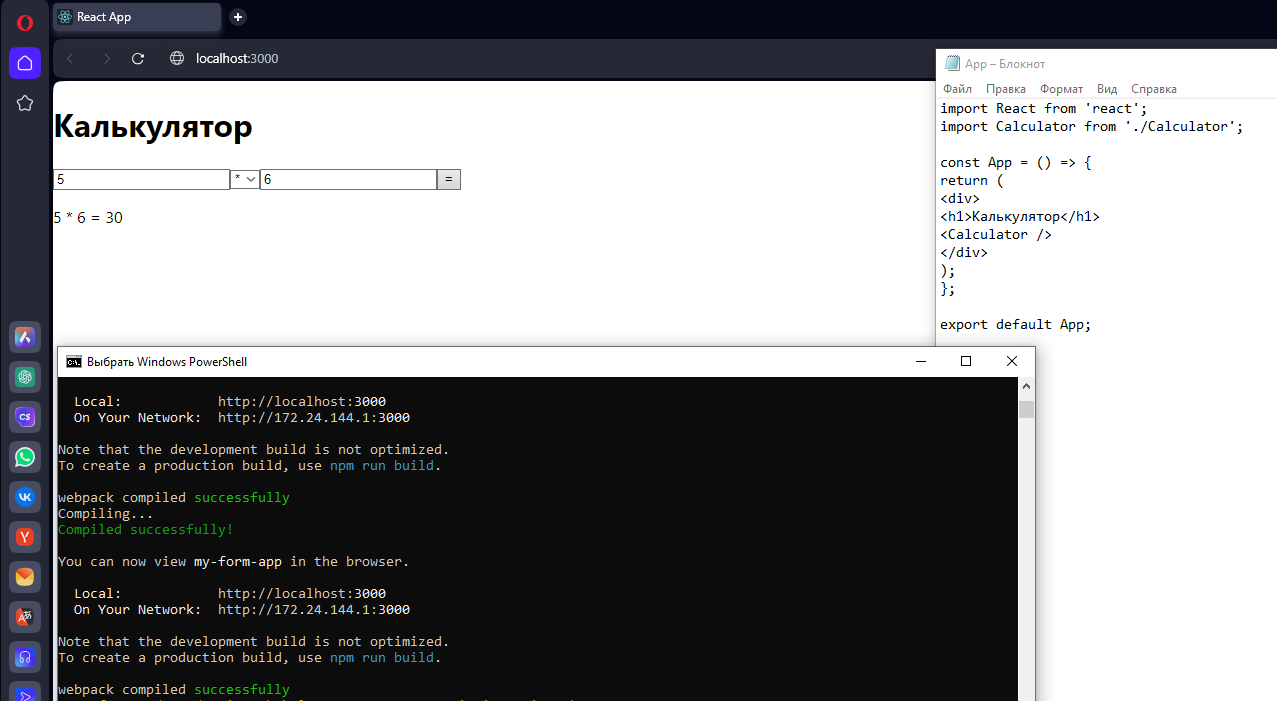
</div>

);

};

export default CitySelector;

**2.- Сделать калькулятор с двумя полями ввода чисел, полем выбора действия и выводом результата текстом рядом: 2 + 2 = 4**



**App.js**

import React from 'react';

import Calculator from './Calculator';

const App = () => {

return (

<div>

<h1>Калькулятор</h1>

<Calculator />

</div>

);

};

export default App;

**Calculator.js**

import React, { useState } from 'react';

const Calculator = () => {

const [num1, setNum1] = useState(0);

const [num2, setNum2] = useState(0);

const [operation, setOperation] = useState('+');

const [result, setResult] = useState(0);

const handleCalculation = () => {

let calculatedResult = 0;

switch (operation) {

case '+':

calculatedResult = num1 + num2;

break;

case '-':

calculatedResult = num1 - num2;

break;

case '\*':

calculatedResult = num1 \* num2;

break;

case '/':

calculatedResult = num1 / num2;

break;

default:

calculatedResult = 0;

}

setResult(calculatedResult);

};

return (

<div>

<input type="number" value={num1} onChange={(e) => setNum1(+e.target.value)} />

<select value={operation} onChange={(e) => setOperation(e.target.value)}>

<option value="+">+</option>

<option value="-">-</option>

<option value="\*">\*</option>

<option value="/">/</option>

</select>

<input type="number" value={num2} onChange={(e) => setNum2(+e.target.value)} />

<button onClick={handleCalculation}>=</button>

<p>{num1} {operation} {num2} = {result}</p>

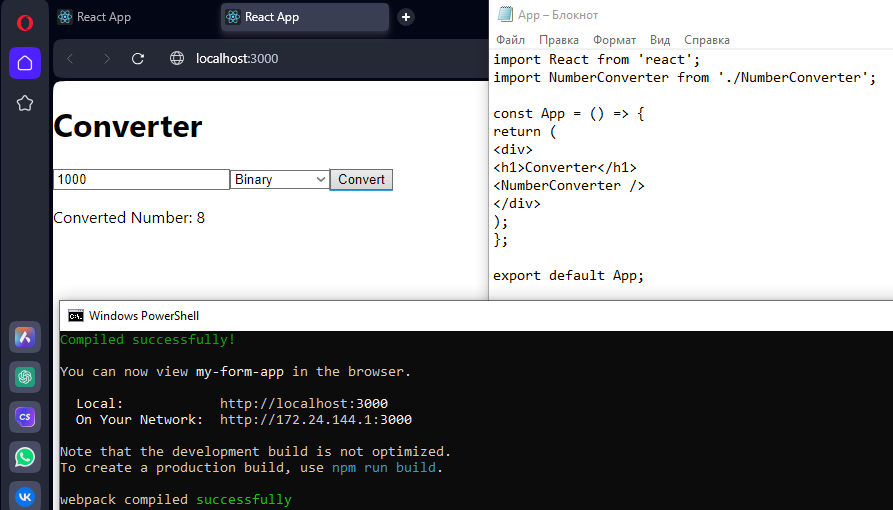
</div>

);

};

export default Calculator;

**3.- Калькулятор систем счисления:15 в 10-чной системе. 1111 в 2 системе.Исходное число вводится в поле ввода, система счисления выбирается из предлложенных, результат отображается во втором поле**



**App.js**

import React from 'react';

import NumberConverter from './NumberConverter';

const App = () => {

return (

<div>

<h1>Converter</h1>

<NumberConverter />

</div>

);

};

export default App;

**NumberConverter.js**

import React, { useState } from 'react';

const NumberConverter = () => {

const [inputNumber, setInputNumber] = useState('');

const [selectedBase, setSelectedBase] = useState(10);

const [convertedNumber, setConvertedNumber] = useState('');

const handleConversion = () => {

const decimalNumber = parseInt(inputNumber, selectedBase);

setConvertedNumber(decimalNumber.toString(10));

};

return (

<div>

<input

type="text"

value={inputNumber}

onChange={(e) => setInputNumber(e.target.value)}

/>

<select value={selectedBase} onChange={(e) => setSelectedBase(+e.target.value)}>

<option value="2">Binary</option>

<option value="8">Octal</option>

<option value="10">Decimal</option>

<option value="16">Hexadecimal</option>

</select>

<button onClick={handleConversion}>Convert</button>

<p>Converted Number: {convertedNumber}</p>

</div>

);

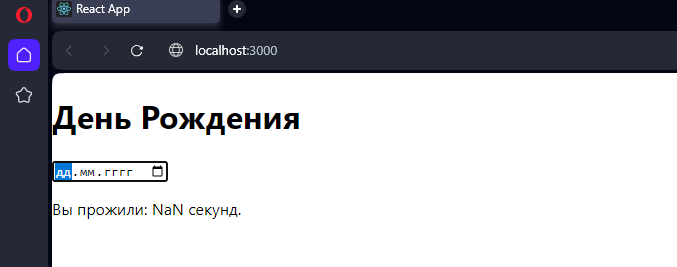
};

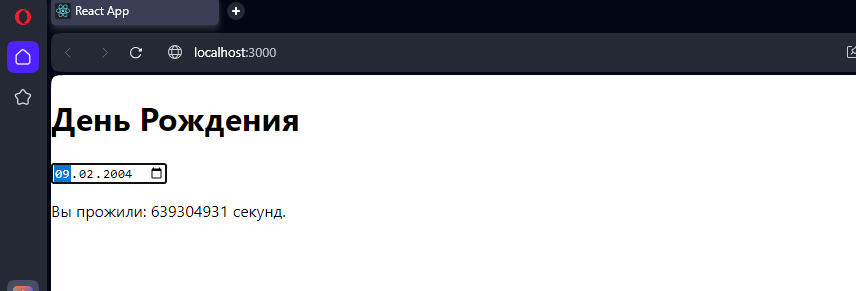
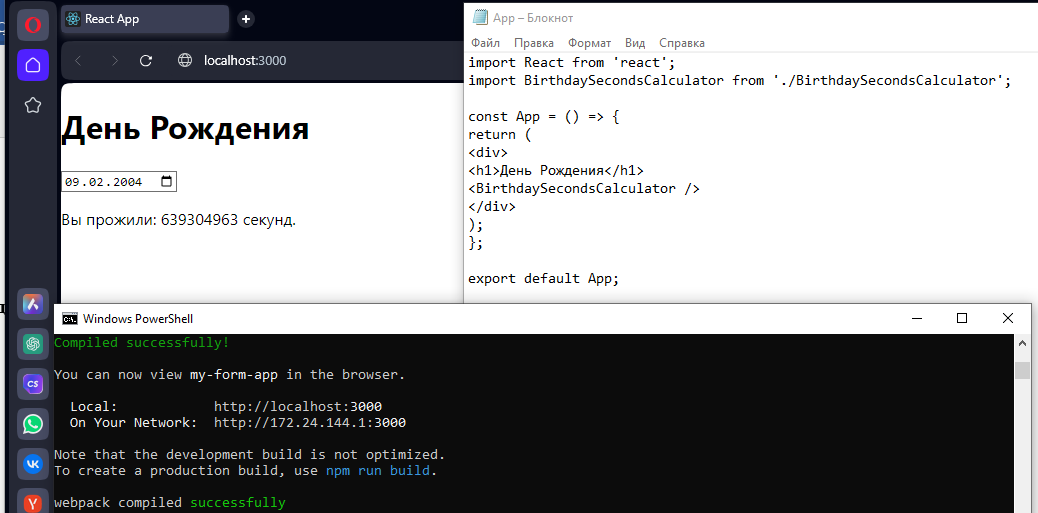
export default NumberConverter;

**4.- В поле ввода выбираем дату рождения: дд.мм.гггг**

**Ниже выводится текстом: "Вы прожили: 1232352345234 секунд."**

**Количество секунд увеличивается каждую секунду.**



**App.js**

import React from 'react';

import BirthdaySecondsCalculator from './BirthdaySecondsCalculator';

const App = () => {

return (

<div>

<h1>День Рождения</h1>

<BirthdaySecondsCalculator />

</div>

);

};

export default App;

**BirthdaySecondsCalculator.js**

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

const BirthdaySecondsCalculator = () => {

const [birthDate, setBirthDate] = useState('');

const [secondsLived, setSecondsLived] = useState(0);

useEffect(() => {

const interval = setInterval(() => {

const birthTime = new Date(birthDate).getTime();

const currentTime = new Date().getTime();

const seconds = Math.floor((currentTime - birthTime) / 1000);

setSecondsLived(seconds);

}, 1000);

return () => clearInterval(interval);

}, [birthDate]);

return (

<div>

<input type="date" value={birthDate} onChange={(e) => setBirthDate(e.target.value)} />

<p>Вы прожили: {secondsLived} секунд.</p>

</div>

);

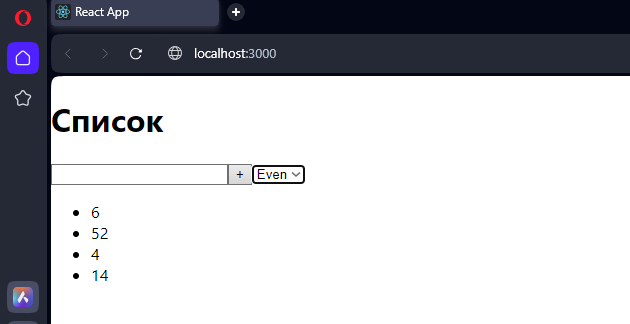
};

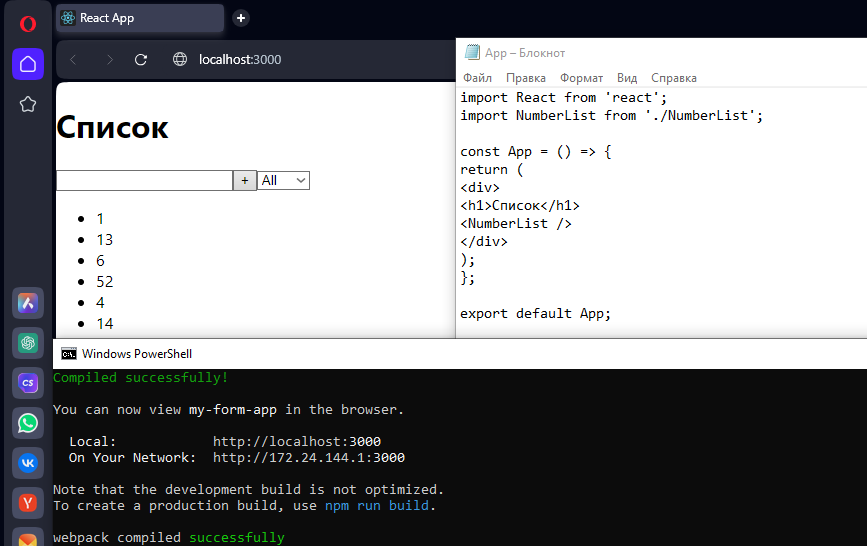
export default BirthdaySecondsCalculator;

**5.Список чисел с фильтрацией: [поле ввода числа] [+] Все Четные Нечетные**

**1, 13, 6, 52, 4, 14**

**При нажатии на [+] в список добавляется очередное число из поля ввода. При изменении состояния фильтра список обновляется.**





**App.js**

import React from 'react';

import NumberList from './NumberList';

const App = () => {

return (

<div>

<h1>Список</h1>

<NumberList />

</div>

);

};

export default App;

**NumberList.js**

import React, { useState } from 'react';

const NumberList = () => {

const [numbers, setNumbers] = useState([1, 13, 6, 52, 4, 14]);

const [newNumber, setNewNumber] = useState('');

const [filter, setFilter] = useState('All');

const addNumber = () => {

if (newNumber !== '') {

setNumbers(prevNumbers => [...prevNumbers, parseInt(newNumber)]);

setNewNumber('');

}

};

const filteredNumbers = numbers.filter(num => {

if (filter === 'Even') {

return num % 2 === 0;

} else if (filter === 'Odd') {

return num % 2 !== 0;

} else {

return true;

}

});

return (

<div>

<input type="number" value={newNumber} onChange={(e) => setNewNumber(e.target.value)} />

<button onClick={addNumber}>+</button>

<select value={filter} onChange={(e) => setFilter(e.target.value)}>

<option value="All">All</option>

<option value="Even">Even</option>

<option value="Odd">Odd</option>

</select>

<ul>

{filteredNumbers.map((num, index) => (

<li key={index}>{num}</li>

))}

</ul>

</div>

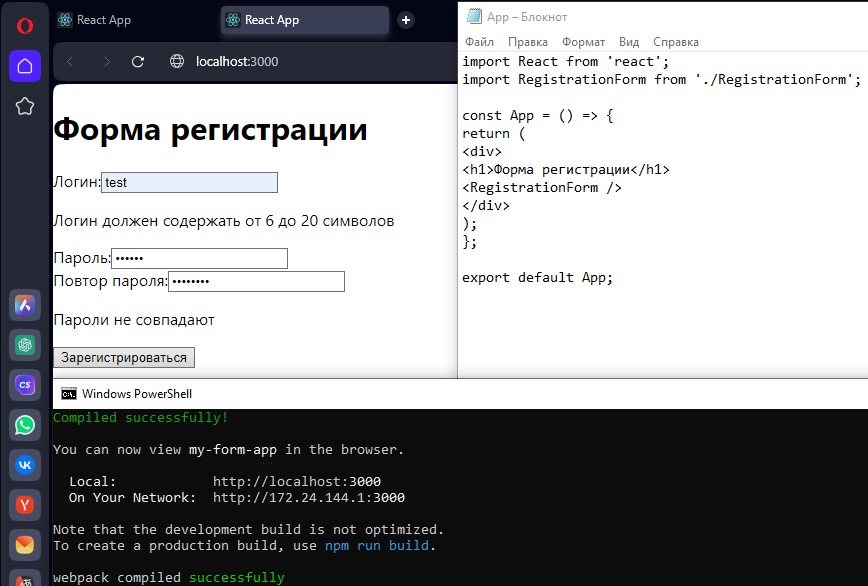
);

};

export default NumberList;

**Задачи про валидацию**

1. **Написать валидацию формы регистрации:**



**App.js**

import React from 'react';

import RegistrationForm from './RegistrationForm';

const App = () => {

return (

<div>

<h1>Форма регистрации</h1>

<RegistrationForm />

</div>

);

};

export default App;

**RegistrationForm.js**

import React, { useState } from 'react';

const RegistrationForm = () => {

const [formData, setFormData] = useState({

username: '',

password: '',

confirmPassword: '',

});

const [errors, setErrors] = useState({

username: '',

password: '',

confirmPassword: '',

});

const validateUsername = (value) => {

if (!value) {

return 'Логин обязателен';

}

if (value.length < 6 || value.length > 20) {

return 'Логин должен содержать от 6 до 20 символов';

}

if (!/^[a-zA-Z0-9]+$/.test(value)) {

return 'Логин может содержать только буквы латинского алфавита и цифры';

}

return '';

};

const validatePassword = (value) => {

if (!value) {

return 'Пароль обязателен';

}

return '';

};

const validateConfirmPassword = (value) => {

if (!value) {

return 'Подтверждение пароля обязательно';

}

if (value !== formData.password) {

return 'Пароли не совпадают';

}

return '';

};

const handleInputChange = (e) => {

const { name, value } = e.target;

setFormData({

...formData,

[name]: value,

});

let error = '';

switch (name) {

case 'username':

error = validateUsername(value);

break;

case 'password':

error = validatePassword(value);

break;

case 'confirmPassword':

error = validateConfirmPassword(value);

break;

default:

break;

}

setErrors({

...errors,

[name]: error,

});

};

const handleSubmit = (e) => {

e.preventDefault();

// Проверка всех полей перед отправкой формы

const usernameError = validateUsername(formData.username);

const passwordError = validatePassword(formData.password);

const confirmPasswordError = validateConfirmPassword(formData.confirmPassword);

setErrors({

username: usernameError,

password: passwordError,

confirmPassword: confirmPasswordError,

});

// Если ошибок нет, можно отправить данные

if (!usernameError && !passwordError && !confirmPasswordError) {

// Отправка данных...

console.log('Данные отправлены:', formData);

}

};

return (

<form onSubmit={handleSubmit}>

<div>

<label>Логин:</label>

<input

type="text"

name="username"

value={formData.username}

onChange={handleInputChange}

/>

{errors.username && <p>{errors.username}</p>}

</div>

<div>

<label>Пароль:</label>

<input

type="password"

name="password"

value={formData.password}

onChange={handleInputChange}

/>

{errors.password && <p>{errors.password}</p>}

</div>

<div>

<label>Повтор пароля:</label>

<input

type="password"

name="confirmPassword"

value={formData.confirmPassword}

onChange={handleInputChange}

/>

{errors.confirmPassword && <p>{errors.confirmPassword}</p>}

</div>

<button type="submit">Зарегистрироваться</button>

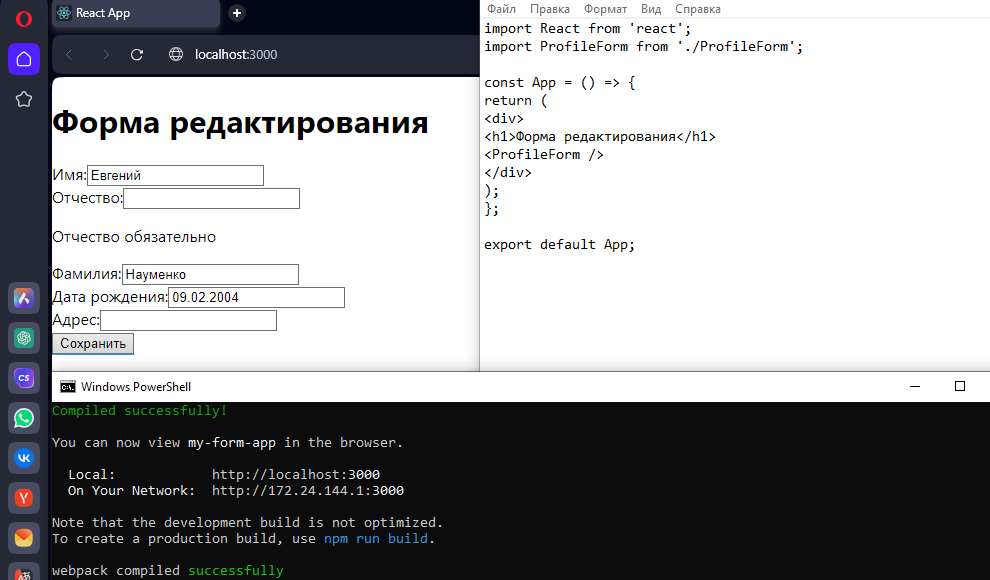
</form>

);

};

export default RegistrationForm;

1. **Написать валидацию формы редактирования профиля:**



**App.js**

import React from 'react';

import ProfileForm from './ProfileForm';

const App = () => {

return (

<div>

<h1>Форма редактирования</h1>

<ProfileForm />

</div>

);

};

export default App;

**ProfileForm.js**

import React, { useState } from 'react';

const ProfileForm = () => {

const [formData, setFormData] = useState({

firstName: '',

middleName: '',

lastName: '',

birthDate: '',

address: '',

});

const [errors, setErrors] = useState({

firstName: '',

middleName: '',

lastName: '',

birthDate: '',

address: '',

});

const validateRequiredField = (name, value) => {

if (!value) {

return `${name} обязательно`;

}

return '';

};

const handleInputChange = (e) => {

const { name, value } = e.target;

setFormData({

...formData,

[name]: value,

});

const error = validateRequiredField(name, value);

setErrors({

...errors,

[name]: error,

});

};

const handleSubmit = (e) => {

e.preventDefault();

// Проверка всех обязательных полей перед отправкой формы

const firstNameError = validateRequiredField('Имя', formData.firstName);

const middleNameError = validateRequiredField('Отчество', formData.middleName);

const lastNameError = validateRequiredField('Фамилия', formData.lastName);

setErrors({

firstName: firstNameError,

middleName: middleNameError,

lastName: lastNameError,

birthDate: '',

address: '',

});

// Если ошибок нет, можно отправить данные

if (!firstNameError && !middleNameError && !lastNameError) {

// Отправка данных...

console.log('Данные отправлены:', formData);

}

};

return (

<form onSubmit={handleSubmit}>

<div>

<label>Имя:</label>

<input

type="text"

name="firstName"

value={formData.firstName}

onChange={handleInputChange}

/>

{errors.firstName && <p>{errors.firstName}</p>}

</div>

<div>

<label>Отчество:</label>

<input

type="text"

name="middleName"

value={formData.middleName}

onChange={handleInputChange}

/>

{errors.middleName && <p>{errors.middleName}</p>}

</div>

<div>

<label>Фамилия:</label>

<input

type="text"

name="lastName"

value={formData.lastName}

onChange={handleInputChange}

/>

{errors.lastName && <p>{errors.lastName}</p>}

</div>

<div>

<label>Дата рождения:</label>

<input

type="text"

name="birthDate"

value={formData.birthDate}

onChange={handleInputChange}

/>

{errors.birthDate && <p>{errors.birthDate}</p>}

</div>

<div>

<label>Адрес:</label>

<input

type="text"

name="address"

value={formData.address}

onChange={handleInputChange}

/>

{errors.address && <p>{errors.address}</p>}

</div>

<button type="submit">Сохранить</button>

</form>

);

};

export default ProfileForm;