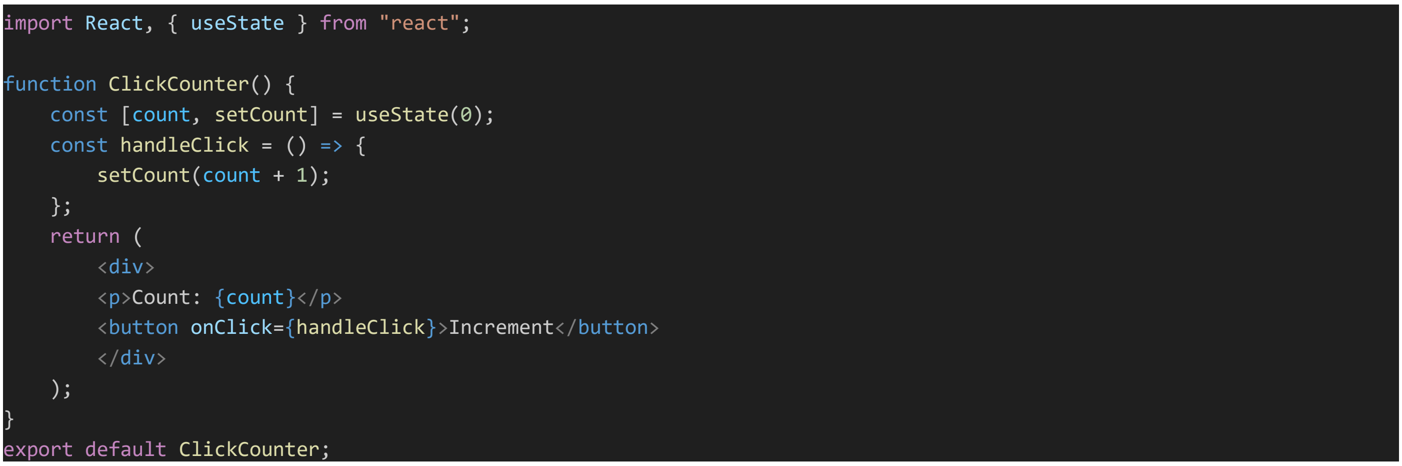
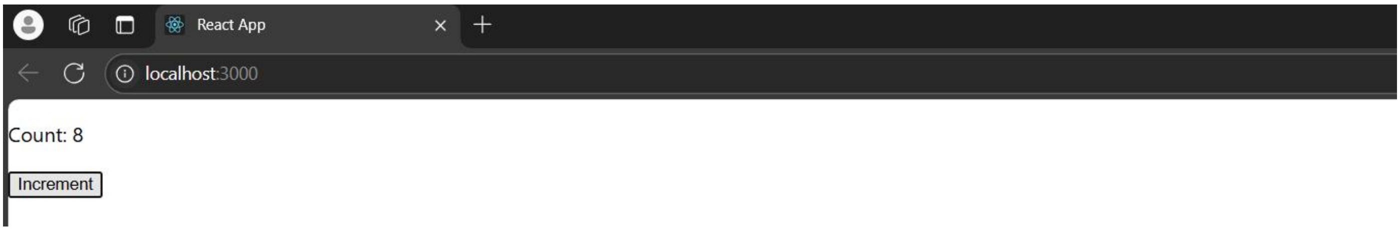
JEYSAN.V
717823F225
MERN TASK day 6 (React)

Exercise 1:Click Counter

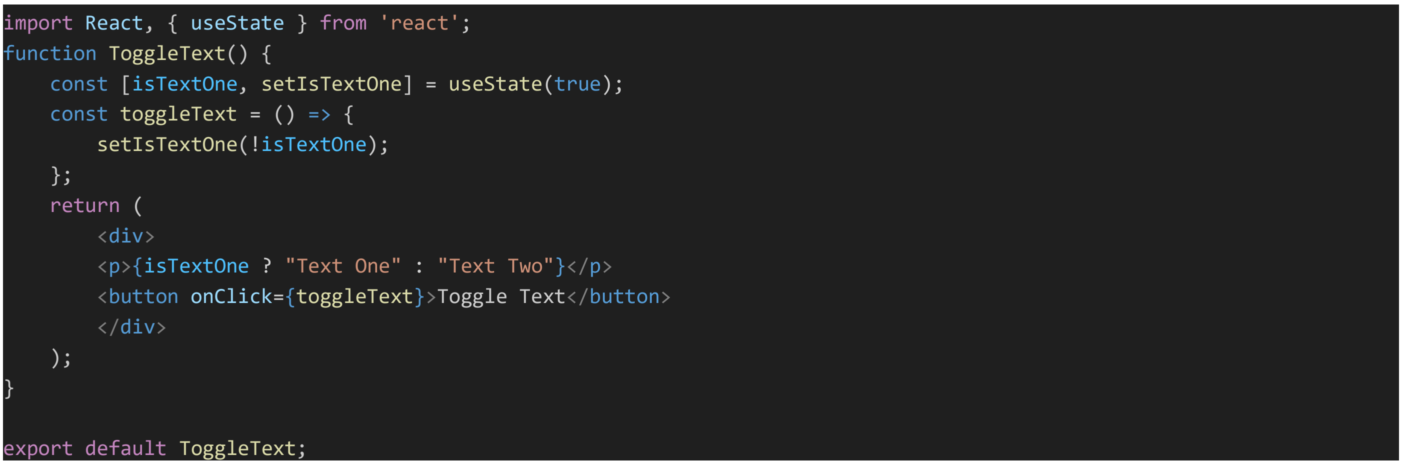


```
import React, { useState } from "react";

function ClickCounter() {
    const [count, setCount] = useState(0);
    const handleClick = () => {
        setCount(count + 1);
    };
    return (
        <div>
        <p>Count: {count}</p>
        <button onClick={handleClick}>Increment</button>
        </div>
    );
}
export default ClickCounter;
```

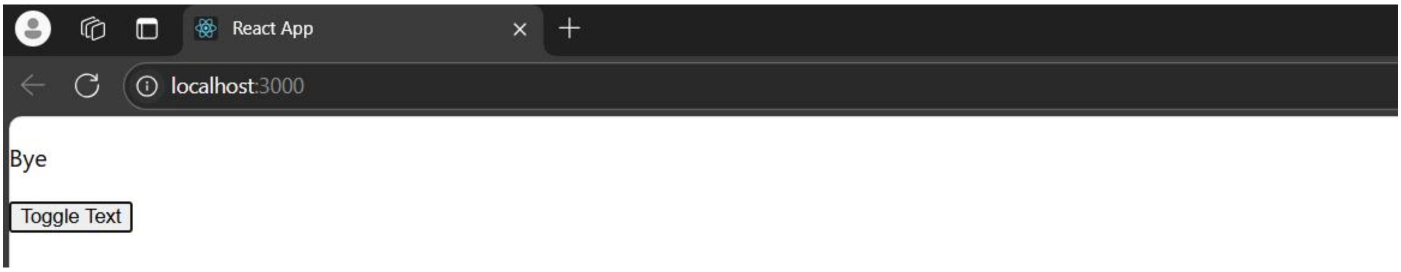


```
Count: 8
```

Exercise 2:Toogle Text

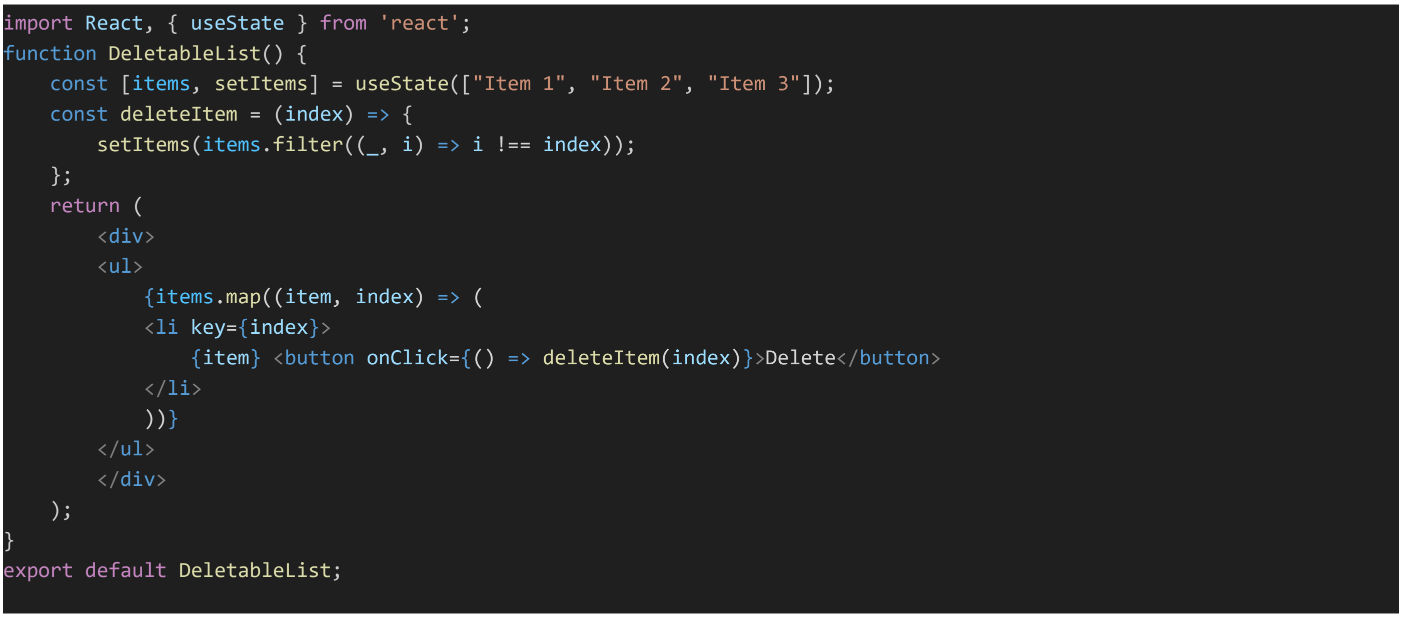


```
import React, { useState } from 'react';
function ToggleText() {
    const [isTextOne, setIsTextOne] = useState(true);
    const toggleText = () => {
        setIsTextOne(!isTextOne);
    };
    return (
        <div>
        <p>{isTextOne ? "Text One" : "Text Two"}</p>
        <button onClick={toggleText}>Toggle Text</button>
        </div>
    );
}

export default ToggleText;
```

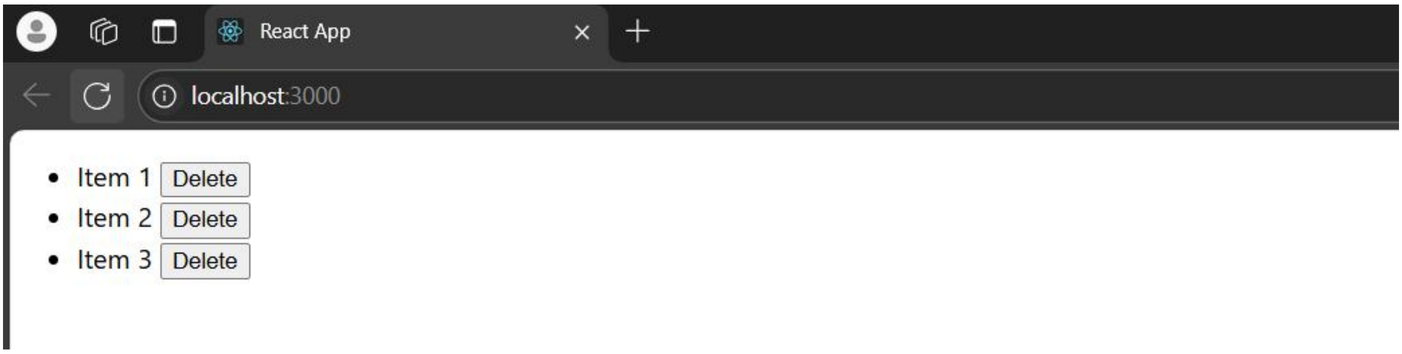


```
Bye
```

Exercise 3:List Item Deletion

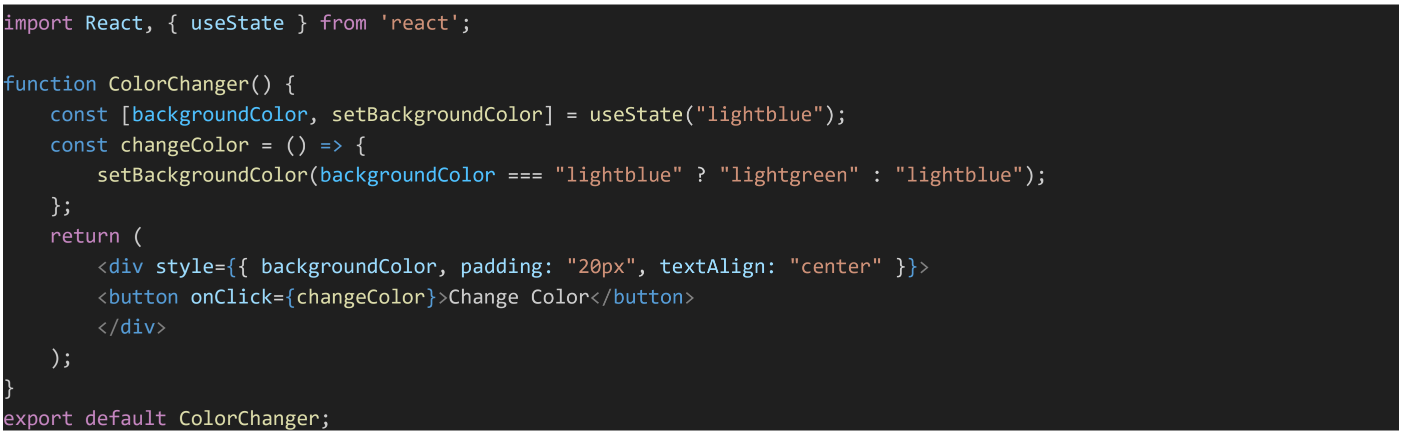


```
import React, { useState } from 'react';
function DeletableList() {
    const [items, setItems] = useState(["Item 1", "Item 2", "Item 3"]);
    const deleteItem = (index) => {
        setItems(items.filter((_, i) => i !== index));
    };
    return (
        <div>
        <ul>
            {items.map((item, index) => (
            <li key={index}>
                {item} <button onClick={() => deleteItem(index)}>Delete</button>
            </li>
            ))}
        </ul>
        </div>
    );
}
export default DeletableList;
```

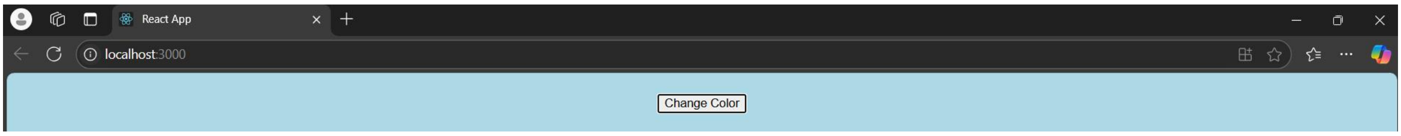


- Item 1 Delete
- Item 2 Delete
- Item 3 Delete

Exercise 4:color Changer

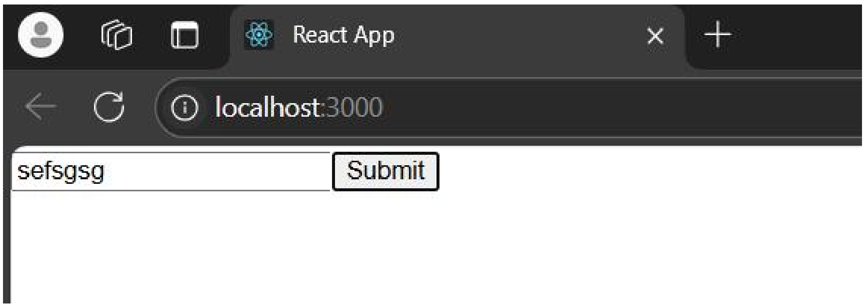


```
import React, { useState } from 'react';

function ColorChanger() {
    const [backgroundColor, setBackgroundColor] = useState("lightblue");
    const changeColor = () => {
        setBackgroundColor(backgroundColor === "lightblue" ? "lightgreen" : "lightblue");
    };
    return (
        <div style={{ backgroundColor, padding: "20px", textAlign: "center" }}>
        <button onClick={changeColor}>Change Color</button>
        </div>
    );
}
export default ColorChanger;
```



Exercise 5:Form Submission

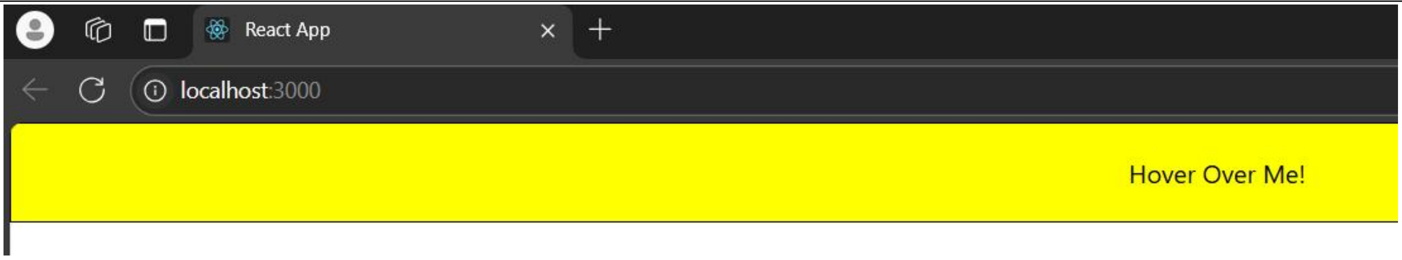
```
import React, { useState } from 'react';

function SimpleForm() {
    const [inputText, setInputText] = useState("");
    const handleInputChange = (e) => {
        setInputText(e.target.value);
    };
    const handleSubmit = (e) => {
        e.preventDefault();
        console.log(inputText);
    };
    return (
        <form onSubmit={handleSubmit}>
        <input
            type="text"
            value={inputText}
            onChange={handleInputChange}
            placeholder="Enter text"
        />
        <button type="submit">Submit</button>
        </form>
    );
}
export default SimpleForm;
```



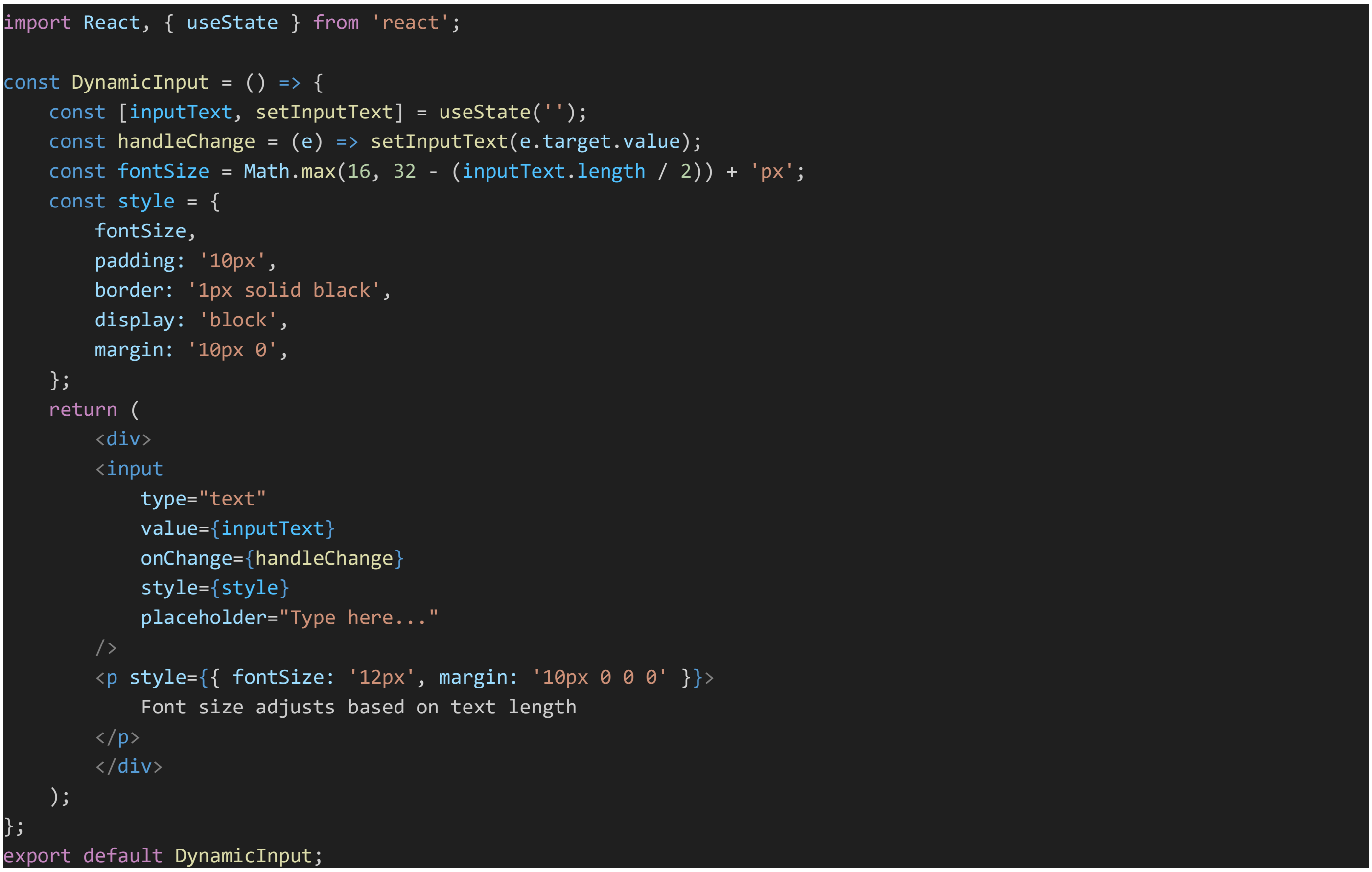
sefsgsg

Exercise 6:Mouse Over Highlighter

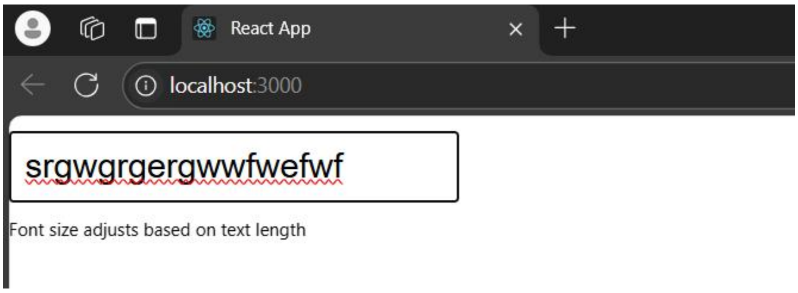
```
import React, { useState } from 'react';

const Highlighter = () => {
    const [isHovered, setIsHovered] = useState(false);
    const handleMouseEnter = () => setIsHovered(true);
    const handleMouseLeave = () => setIsHovered(false);
    const style = {
        backgroundColor: isHovered ? 'yellow' : 'transparent',
        padding: '20px',
        textAlign: 'center',
        border: '1px solid black',
    };
    return (
        <div
        style={style}
        onMouseEnter={handleMouseEnter}
        onMouseLeave={handleMouseLeave}
        >
        Hover Over Me!
        </div>
    );
};
export default Highlighter;
```



Exercise 7:Dynamic Input Field



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

const DynamicInput = () => {
    const [inputText, setInputText] = useState('');
    const handleChange = (e) => setInputText(e.target.value);
    const fontSize = Math.max(16, 32 - (inputText.length / 2)) + 'px';
    const style = {
        fontSize,
        padding: '10px',
        border: '1px solid black',
        display: 'block',
        margin: '10px 0',
    };
    return (
        <div>
        <input
            type="text"
            value={inputText}
            onChange={handleChange}
            style={style}
            placeholder="Type here..."
        />
        <p style={{ fontSize: '12px', margin: '10px 0 0 0' }}>
            Font size adjusts based on text length
        </p>
        </div>
    );
};
export default DynamicInput;
```

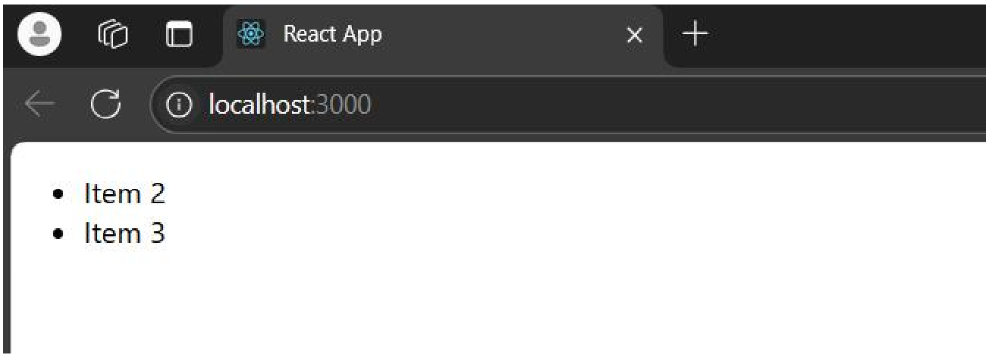


srgwgrgergwwfwefwf

Font size adjusts based on text length

Exercise 8:Double Click to Remove

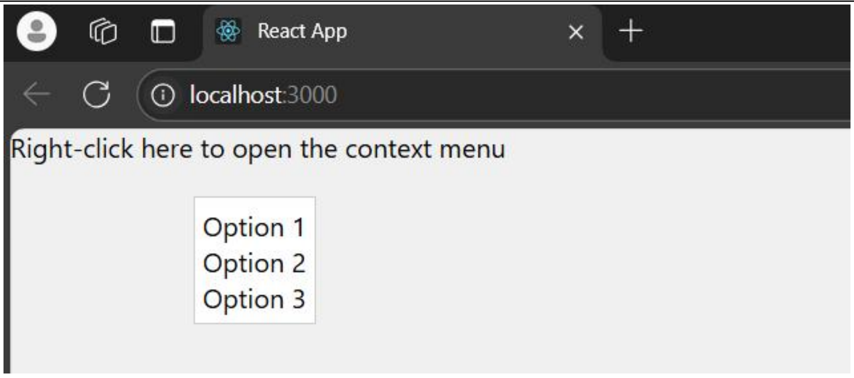
```
import React, { useState } from 'react';

const DoubleClickRemove = () => {
    const [items, setItems] = useState(['Item 1', 'Item 2', 'Item 3']);
    const handleDoubleClick = (index) => {
        const newItems = items.filter((_, i) => i !== index);
        setItems(newItems);
    };
    return (
        <ul>
        {items.map((item, index) => (
            <li key={index} onDoubleClick={() => handleDoubleClick(index)}>
            {item}
            </li>
        ))}
        </ul>
    );
};
export default DoubleClickRemove;
```



- Item 2
- Item 3

Exercise 9:Right Click Menu

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

const ContextMenu = () => {
    const [menuVisible, setMenuVisible] = useState(false);
    const [position, setPosition] = useState({ x: 0, y: 0 });
    const handleContextMenu = (event) => {
        event.preventDefault();
        setPosition({ x: event.pageX, y: event.pageY });
        setMenuVisible(true);
    };
    const handleClick = () => setMenuVisible(false);
    return (
        <div onContextMenu={handleContextMenu} onClick={handleClick} style={{ height: '100vh', background:
'#f0f0f0' }}>
        Right-click here to open the context menu
        {menuVisible && (
            <ul style={{ position: 'absolute', top: position.y, left: position.x, listStyle: 'none', background:
'#fff', padding: '5px', border: '1px solid #ccc' }}>
            <li>Option 1</li>
            <li>Option 2</li>
            <li>Option 3</li>
            </ul>
        )}
        </div>
    );
};
export default ContextMenu;
```

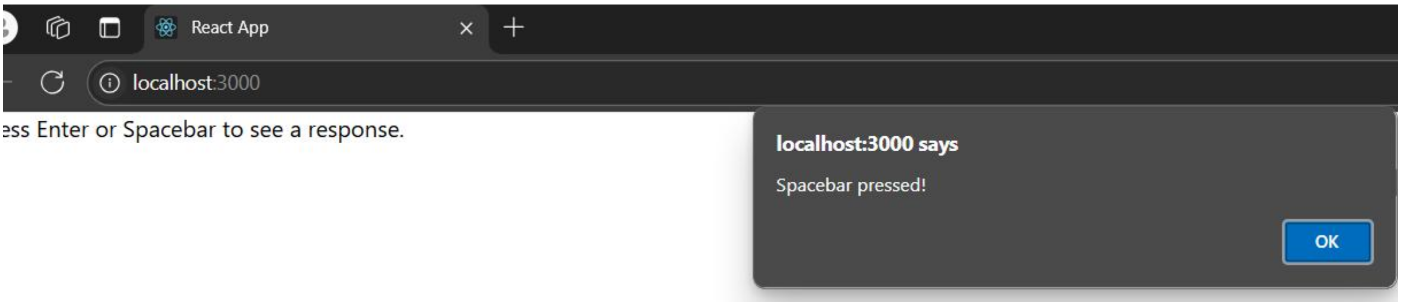


Right-click here to open the context menu

Option 1
Option 2
Option 3

Exercise 10:Keyboard Event

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

const KeyboardListener = () => {
    const handleKeyDown = (event) => {
        if (event.keyCode === 13) {
        alert('Enter key pressed!');
        } else if (event.keyCode === 32) {
        alert('Spacebar pressed!');
        }
    };
    useEffect(() => {
        window.addEventListener('keydown', handleKeyDown);
        return () => {
        window.removeEventListener('keydown', handleKeyDown);
        };
    }, []);
    return (
        <div>
        Press Enter or Spacebar to see a response.
        </div>
    );
};
export default KeyboardListener;
```



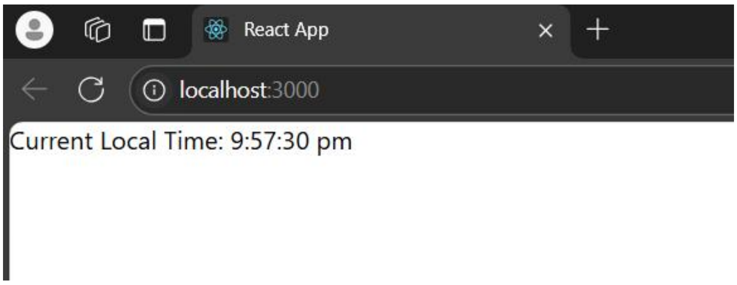
ess Enter or Spacebar to see a response.

localhost:3000 says

Spacebar pressed!

Exercise 11:Component Memory

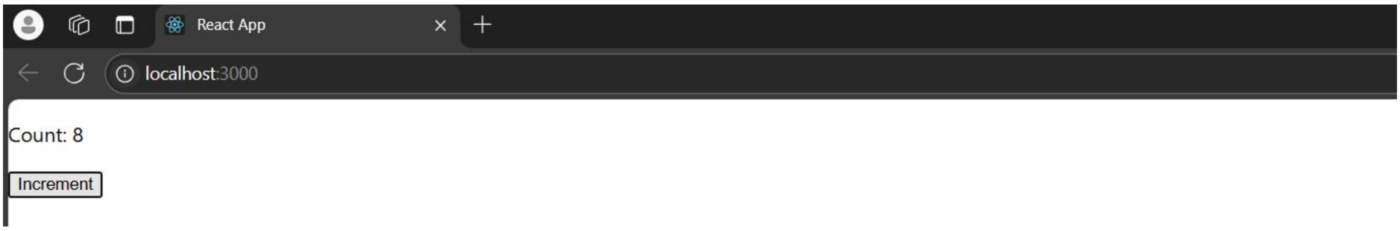
```
import React, { useState, useEffect } from 'react';

const LocalTime = () => {
    const [currentTime, setCurrentTime] = useState(new Date());
    useEffect(() => {
        const timer = setInterval(() => {
        setCurrentTime(new Date());
        }, 1000);
        return () => {
        clearInterval(timer);
        };
    }, []);
    return (
        <div>
        Current Local Time: {currentTime.toLocaleTimeString()}
        </div>
    );
};
export default LocalTime;
```



Current Local Time: 9:57:30 pm

Exercise 12:Counter with Reset

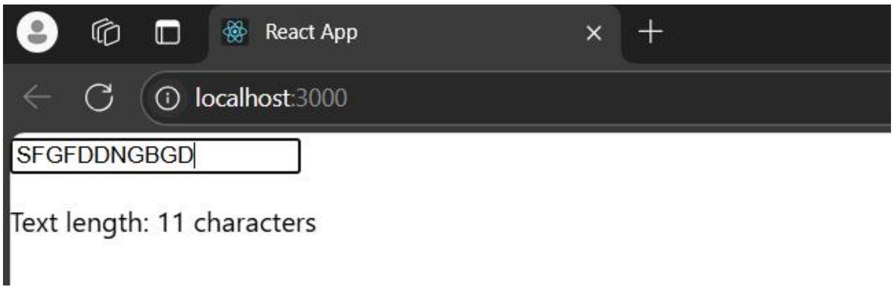
```
import React, { useState } from "react";

function ClickCounter() {
    const [count, setCount] = useState(0);
    const handleClick = () => {
        setCount(count + 1);
    };
    return (
        <div>
        <p>Count: {count}</p>
        <button onClick={handleClick}>Increment</button>
        </div>
    );
}
export default ClickCounter;
```



Count: 8

Exercise 13:Text Length indicator

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

const TextLengthIndicator = () => {
    const [text, setText] = useState('');
    const handleChange = (event) => {
        setText(event.target.value);
    };
    return (
        <div>
        <input
            type="text"
            value={text}
            onChange={handleChange}
            placeholder="Type something..."
        />
        <p>Text length: {text.length} characters</p>
        </div>
    );
};
export default TextLengthIndicator;
```

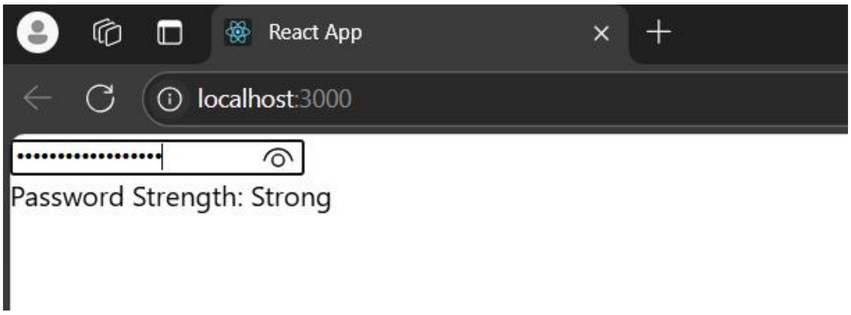


SFGFDDNGBGD

Text length: 11 characters

Exercise 14:Password Strength

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

const PasswordStrength = () => {
    const [password, setPassword] = useState('');
    const [strength, setStrength] = useState('');
    const evaluateStrength = (password) => {
        let strength = '';
        if (password.length >= 8) {
        strength = 'Strong';
        } else if (password.length >= 5) {
        strength = 'Medium';
        } else {
        strength = 'Weak';
        }
        return strength;
    };
    const handleChange = (e) => {
        const newPassword = e.target.value;
        setPassword(newPassword);
        setStrength(evaluateStrength(newPassword));
    };
    return (
        <div>
        <input
            type="password"
            value={password}
            onChange={handleChange}
            placeholder="Enter your password"
        />
        <div>Password Strength: {strength}</div>
        </div>
    );
};
export default PasswordStrength;
```

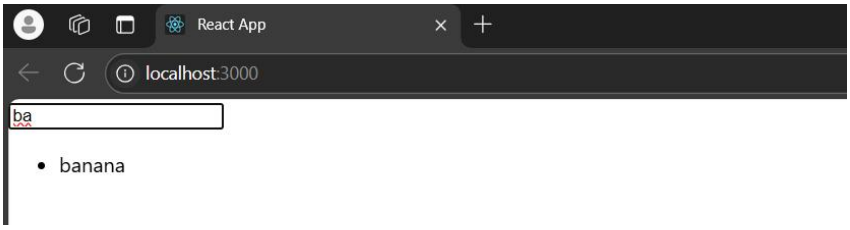


•••••••••••••••••

Password Strength: Strong

Exercise 15:Autocomplete drop down

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

const AutoComplete = () => {
    const [input, setInput] = useState('');
    const [suggestions, setSuggestions] = useState([]);
    const possibleCompletions = ['apple', 'banana', 'grape', 'orange', 'strawberry'];
    const handleChange = (e) => {
        const text = e.target.value;
        setInput(text);
        if (text.length > 0) {
        const filteredSuggestions = possibleCompletions.filter((item) =>
            item.toLowerCase().includes(text.toLowerCase())
        );
        setSuggestions(filteredSuggestions);
        } else {
        setSuggestions([]);
        }
    };
    return (
        <div>
        <input
            type="text"
            value={input}
            onChange={handleChange}
            placeholder="Start typing..."
        />
        {suggestions.length > 0 && (
            <ul>
            {suggestions.map((suggestion, index) => (
                <li key={index}>{suggestion}</li>
            ))}
            </ul>
        )}
        </div>
    );
};
export default AutoComplete;
```



ba

- banana