**Practical No.: 07**

**Aim: Using useRef Hook**

1. Develop a React application that includes a form with input fields for username and email address. Implement functionality to automatically focus on the username input field when the page loads.

import React, { useRef } from 'react';

function FocusRef() {

const inputRef = useRef(null);

const focusInput = () => {

inputRef.current.focus();

};

return (

<div>

<h1>Focus Input Field</h1>

<input type="text" ref={inputRef} />

<button onClick={focusInput}>Focus Input</button>

</div>

);

}

export default FocusRef

**Output:**

A screenshot of a computer

Description automatically generated

1. Develop a React component called FormWithRef that implements a form with input fields for username and email. Use the useRef hook to persist mutable values for these input fields and to store the submitted data. The form should log the submitted data to the console when the user submits the form.

import React, { useRef } from 'react';

function P7ii() {

const usernameRef = useRef('');

const emailRef = useRef('');

const submittedDataRef = useRef(null);

const handleSubmit = (event) => {

event.preventDefault();

const username = usernameRef.current.value;

const email = emailRef.current.value;

submittedDataRef.current = { username, email };

usernameRef.current.value = '';

emailRef.current.value = '';

console.log('Submitted Data:', submittedDataRef.current);

};

return (

<div>

<h2>Form with Ref</h2>

<form onSubmit={handleSubmit}>

<div>

<label htmlFor="username">Username:</label>

<input

type="text"

id="username"

ref={usernameRef}

/>

</div>

<div>

<label htmlFor="email">Email:</label>

<input

type="email"

id="email"

ref={emailRef}

/>

</div>

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

</form>

</div>

);

}

export default P7ii;

**Output:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

1. Develop a React application that demonstrates parent-child interaction using the useImperativeHandle hook. Create a parent component and a child component. The child component should have a method that, when called by the parent component, displays a message.

import React, { useRef, useImperativeHandle, forwardRef } from 'react';

const ChildComponent = forwardRef((props, ref) => {

const greetParent = () => {

console.log('Hello from Child Component!');

};

useImperativeHandle(ref, () => ({

greetParent: greetParent

}));

return (

<div>

<h3>Child Component</h3>

</div>

);

});

function P7iii() {

const childRef = useRef(null);

const handleButtonClick = () => {

childRef.current.greetParent();

};

return (

<div>

<h2>Parent Component</h2>

<button onClick={handleButtonClick}>Greet Child</button>

<ChildComponent ref={childRef} />

</div>

);

}

export default P7iii;

**Output:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**