Handson Only:

1. Build a Simple Counter

- **Task**: Create a counter that increments, decrements, and resets to zero when buttons are clicked.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const Counter = () => {
  const [count, setCount] = useState(0);

  return (
    <div>
      <h2>Counter: {count}</h2>
      <button onClick={() => setCount(count + 1)}>Increment</
button>
      <button onClick={() => setCount(count - 1)}>Decrement</
button>
      <button onClick={() => setCount(0)}>Reset</button>
    </div>
  );
};

export default Counter;
```

- **Core Concepts**: `useState`, event handling for buttons.

2. Build a To-Do List

- **Task**: Create a to-do list where users can add tasks and delete them.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const TodoApp = () => {
  const [task, setTask] = useState('');
  const [tasks, setTasks] = useState([]);

  const addTask = () => {
    setTasks([...tasks, task]);
    setTask('');
  };

  const deleteTask = (index) => {
    setTasks(tasks.filter((_, i) => i !== index));
  };

  return (
    <div>
      <input
        type="text"
        value={task}
        onChange={(e) => setTask(e.target.value)}
        placeholder="Enter task"
      />
      <button onClick={addTask}>Add Task</button>

      <ul>
        {tasks.map((task, index) => (
```

```
          <li key={index}>
            {task} <button onClick={() => deleteTask(index)}>
Delete</button>
          </li>
        ))}
      </ul>
    </div>
  );
};

export default TodoApp;
```

- **Core Concepts**: `useState`, list rendering, handling user input.

---

3. Create a Basic Form with Validation

- **Task**: Create a form with **name** and **email** fields and basic validation.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const FormValidation = () => {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [error, setError] = useState('');

  const handleSubmit = (e) => {
    e.preventDefault();
    if (!name || !email || !/\S+@\S+\.\S+/.test(email)) {
      setError('Please fill out all fields with valid dat
a.');
    } else {
```

```
      setError('');
      alert('Form submitted successfully');
    }
  };

  return (
    <div>
      <h2>Form</h2>
      <form onSubmit={handleSubmit}>
        <input
          type="text"
          placeholder="Name"
          value={name}
          onChange={(e) => setName(e.target.value)}
        />
        <input
          type="email"
          placeholder="Email"
          value={email}
          onChange={(e) => setEmail(e.target.value)}
        />
        <button type="submit">Submit</button>
      </form>
      {error && <p style={{ color: 'red' }}>{error}</p>}
    </div>
  );
};

export default FormValidation;
```

- **Core Concepts**: `useState`, form validation, conditional rendering.

4. Build a Simple Toggle Button (Light/Dark Theme)

- **Task**: Create a button that toggles between light and dark themes.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const ThemeToggle = () => {
  const [isDark, setIsDark] = useState(false);

  const toggleTheme = () => {
    setIsDark(!isDark);
  };

  return (
    <div style={{ backgroundColor: isDark ? '#333' : '#fff',
color: isDark ? '#fff' : '#000' }}>
      <h2>Theme Toggle</h2>
      <button onClick={toggleTheme}>
        Switch to {isDark ? 'Light' : 'Dark'} Mode
      </button>
    </div>
  );
};

export default ThemeToggle;
```

- **Core Concepts**: `useState`, conditional styling.

---

5. Create a Search Filter

- **Task**: Create a list of items and a search bar that filters the list based on user input.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const SearchFilter = () => {
  const [query, setQuery] = useState('');
  const items = ['Apple', 'Banana', 'Orange', 'Mango', 'Grape
s'];

  const filteredItems = items.filter(item => item.toLowerCase
().includes(query.toLowerCase()));

  return (
    <div>
      <input
        type="text"
        placeholder="Search..."
        value={query}
        onChange={(e) => setQuery(e.target.value)}
      />
      <ul>
        {filteredItems.map((item, index) => (
          <li key={index}>{item}</li>
        ))}
      </ul>
    </div>
  );
};

export default SearchFilter;
```

- **Core Concepts**: `useState`, filtering arrays, handling user input.

6. Build a Simple Timer

- **Task**: Create a countdown timer that starts from a given number and decreases every second.

Solution:

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

const Timer = () => {
  const [time, setTime] = useState(10);

  useEffect(() => {
    if (time > 0) {
      const timerId = setInterval(() => setTime(time - 1), 10
00);
      return () => clearInterval(timerId);
    }
  }, [time]);

  return (
    <div>
      <h2>Countdown Timer: {time}s</h2>
      {time === 0 && <p>Time's up!</p>}
    </div>
  );
};

export default Timer;
```

- **Core Concepts**: `useState`, `useEffect`, `setInterval`.

---

7. Create a Simple Modal

- **Task**: Create a modal that opens when a button is clicked and closes when the close button is clicked.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const Modal = () => {
  const [isOpen, setIsOpen] = useState(false);

  const toggleModal = () => setIsOpen(!isOpen);

  return (
    <div>
      <button onClick={toggleModal}>Open Modal</button>
      {isOpen && (
        <div style={{ background: 'rgba(0, 0, 0, 0.5)', paddi
ng: '20px' }}>
          <div style={{ background: 'white', padding: '20px'
}}>
            <h2>Modal Content</h2>
            <button onClick={toggleModal}>Close Modal</button
>
          </div>
        </div>
      )}
    </div>
  );
};

export default Modal;
```

- **Core Concepts**: `useState`, conditional rendering for modal visibility.

8. Build a Simple Image Gallery

- **Task**: Create a gallery with clickable thumbnails that show the image in full size in a modal.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const ImageGallery = () => {
  const [selectedImage, setSelectedImage] = useState(null);
  const images = ['img1.jpg', 'img2.jpg', 'img3.jpg'];

  const openImage = (image) => setSelectedImage(image);
  const closeImage = () => setSelectedImage(null);

  return (
    <div>
      <h2>Image Gallery</h2>
      <div style={{ display: 'flex' }}>
        {images.map((image, index) => (
          <img
            key={index}
            src={image}
            alt={`Thumbnail ${index}`}
            style={{ width: 100, marginRight: 10 }}
            onClick={() => openImage(image)}
          />
        ))}
      </div>
      {selectedImage && (
        <div style={{ padding: '20px', background: 'rgba(0,
0, 0, 0.5)' }}>
```

```
          <div style={{ background: 'white', padding: '20px'
}}>
            <img src={selectedImage} alt="Full Size" style={{
width: '100%' }} />
            <button onClick={closeImage}>Close</button>
          </div>
        </div>
      )}
    </div>
  );
};

export default ImageGallery;
```

- **Core Concepts**: `useState`, modal management, image rendering.

---

9. Create a Simple Accordion

- **Task**: Create an accordion where users can click to expand or collapse sections.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const Accordion = () => {
  const [openIndex, setOpenIndex] = useState(null);

  const toggleSection = (index) => {
    setOpenIndex(openIndex === index ? null : index);
  };

  return (
```

```
    <div>
      <h2>Accordion</h2>
      {['Section 1', 'Section 2', 'Section 3'].map((section,
index) => (
        <div key={index}>
          <button onClick={() => toggleSection(index)}>{secti
on}</button>
          {openIndex === index && <p>This is the content of
{section}.</p>}
        </div>
      ))}
    </div>
  );
};

export default Accordion;
```

- **Core Concepts**: `useState`, toggling content visibility, event handling.

---

10. Build a Rating Component

- **Task**: Create a rating component with stars where users can click on stars to give a rating.

Solution:

```
jsx
Copy code
import React, { useState } from 'react';

const Rating = () => {
  const [rating, setRating] = useState(0);

  const handleRating = (rate) => {
    setRating(rate);
```

```
  };

  return (
    <div>
      <h2>Rate this product</h2>
      {[1, 2, 3, 4, 5].map((rate) => (
        <span
          key={rate}
          style={{ cursor: 'pointer', color: rate <= rating ?
'gold' : 'gray' }}
          onClick={() => handleRating(rate)}
        >
          &#9733;
        </span>
      ))}
      <p>You rated: {rating} stars</p>
    </div>
  );
};

export default Rating;
```

- **Core Concepts**: `useState`, event handling, visual feedback (star rating).