

MERN Stack Training

Weekly Tasks

Week 4:

1. How to write your first React component Tasks:

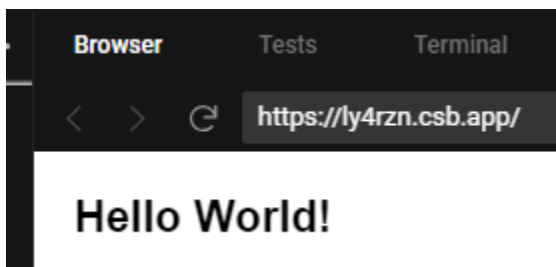
Tasks:

1. Create a simple functional component named Greeting that returns “Hello, World!” within a tag.

Code:

```
function MyApp() {  
  return (  
    <div>  
      <h1>Hello World!</h1>  
    </div>  
  );  
}  
export default MyApp;
```

Output:



2.Modify the Greeting component to display “Hello, React!”.

Code:

```
function MyApp() {  
  return (  
    <div>  
      <h1>“Hello, React!”</h1>  
    </div>  
  );  
}  
export default MyApp;
```

Output:

“Hello, React!”

3.Create a Gallery functional component to display an image.

Code:

```
function Picture() {  
  return (  
    <div>  
        
      </div>  
    );  
}  
export default Picture;
```

Output:



4. Add Greeting to the Gallery component and display the image and greeting.

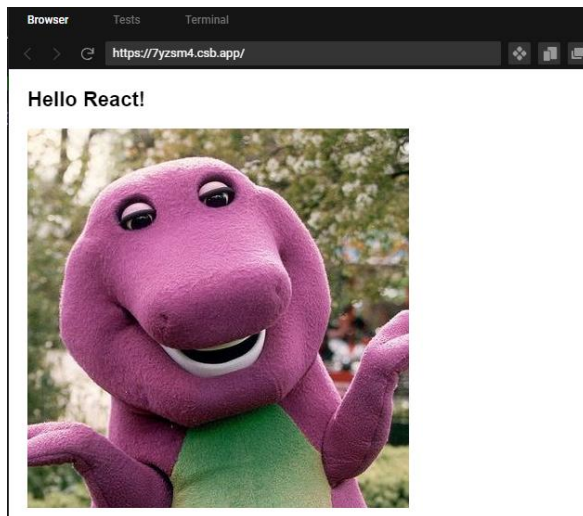
Code:

```
function Picture() {
  return (
    <div>
      
      </div>
    );
}

function Gallery() {
  return (
    <section>
      <h1>Hello React!</h1>
      <Picture />
    </section>
  );
}
```

```
);  
}  
export default Gallery;
```

Output:

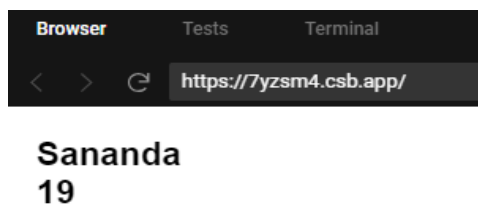


5. Write a component called Profile which displays a hardcoded user's name and age.

Code:

```
function Profile() {  
  return (  
    <section>  
      <h1>Sananda<br></br>19</h1>  
    </section>  
  );  
}  
export default Profile;
```

Output:

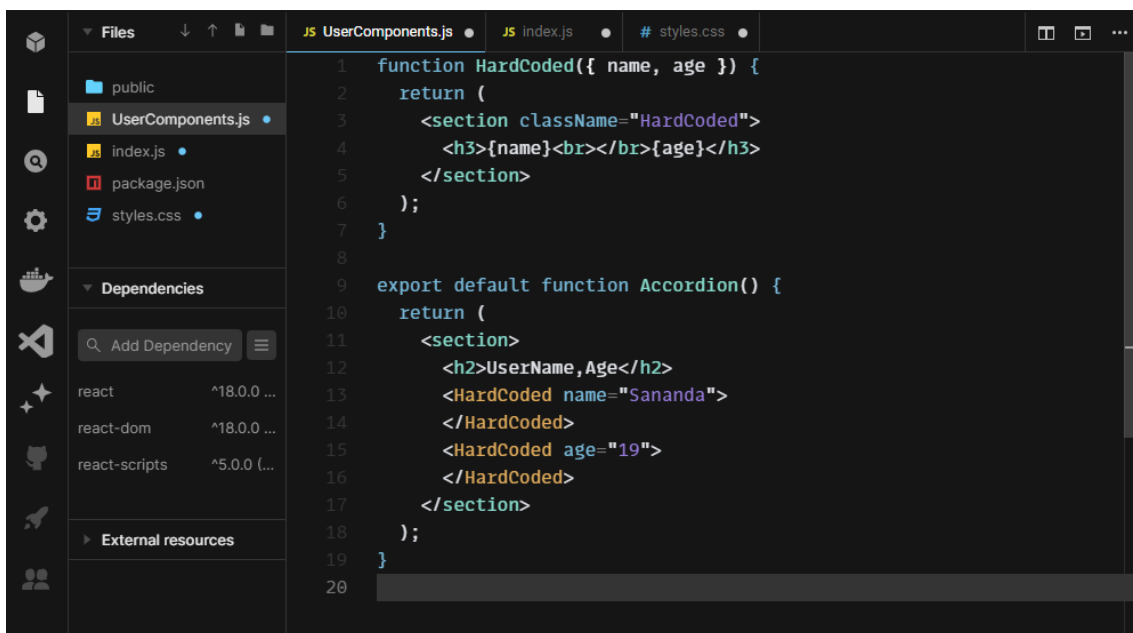


2. When and how to create multi-component files

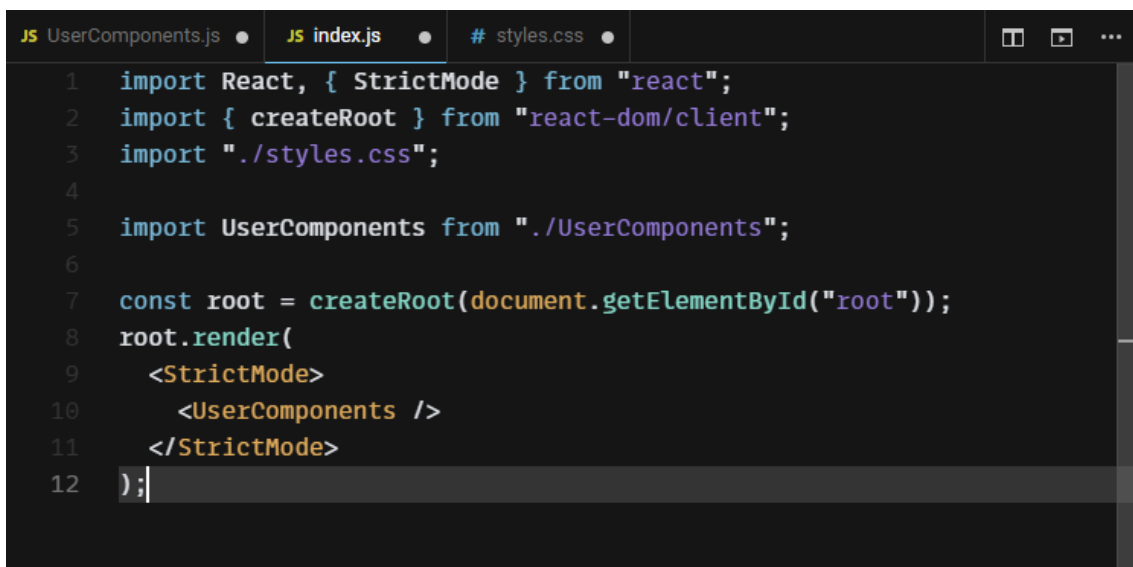
Tasks:

1. Create a file named UserComponents.js and inside it, define two components: UserName and UserAge that display hardcoded names and ages respectively.
2. Export both UserName and UserAge from UserComponents.js.

Code:

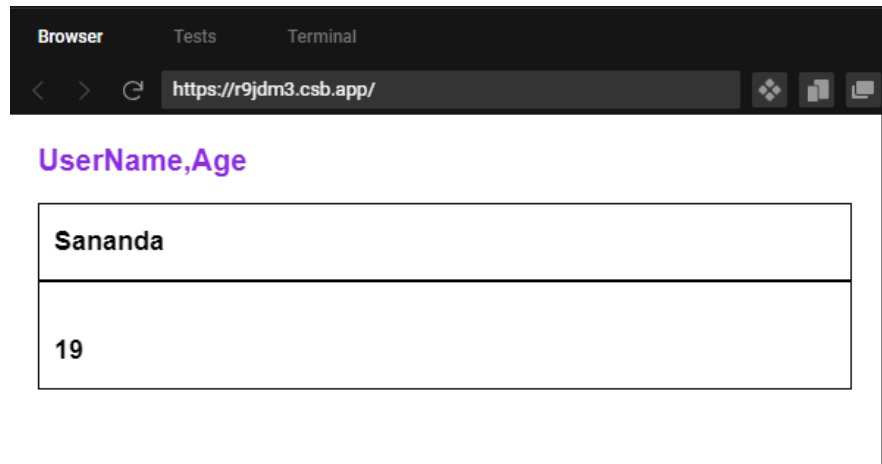


```
1 function HardCoded({ name, age }) {
2   return (
3     <section className="HardCoded">
4       <h3>{name}<br/>{age}</h3>
5     </section>
6   );
7 }
8
9 export default function Accordion() {
10   return (
11     <section>
12       <h2>UserName, Age</h2>
13       <HardCoded name="Sananda">
14     </HardCoded>
15       <HardCoded age="19">
16     </HardCoded>
17     </section>
18   );
19 }
20
```



```
1 import React, { StrictMode } from "react";
2 import { createRoot } from "react-dom/client";
3 import "./styles.css";
4
5 import UserComponents from "./UserComponents";
6
7 const root = createRoot(document.getElementById("root"));
8 root.render(
9   <StrictMode>
10     <UserComponents />
11   </StrictMode>
12 );
```

Output:



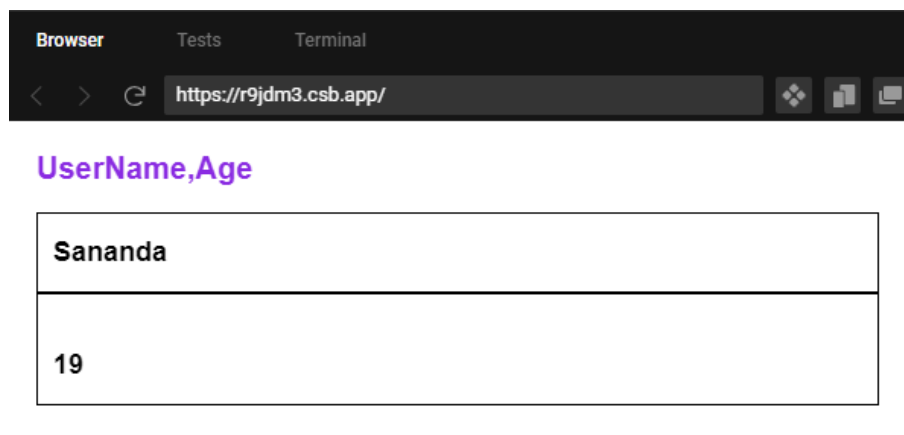
3. In a separate file, import and use both UserName and UserAge components using named imports.

Code:

```
import Accordion from './importValues.js';

export default function UserComponents() {
  return (
    <Accordion />
  );
}
```

Output:



4. Convert UserAge into a default export and modify the importing file to accommodate the change.
5. Split UserName and UserAge into separate files and adjust your imports

Code:

```
// UserName.js
import React from 'react';

export class UserName extends React.Component {
  render() {
    return (
      <div>
        <h1>Name : Sana</h1>
      </div>
    );
  }
}

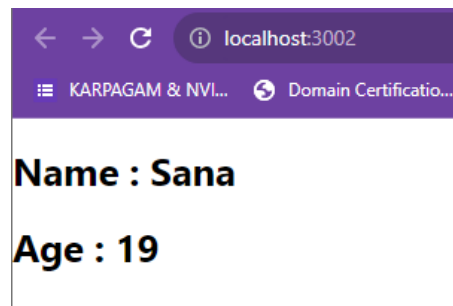
// UserAge.js
import React from 'react';

class UserAge extends React.Component {
  render() {
    return (
      <div>
        <h1>Age : 19</h1>
      </div>
    );
  }
}

export default UserAge;
```

```
VS Code: Welcome | JS App.js M X | JS UserName.js U | JS UserAge.js U | JS index.js M
first > src > JS App.js > [default]
1  import './App.css';
2  import React from 'react';
3  import {UserName} from './UserName'; // Importing UserName from a separate file
4  import UserAge from './UserAge'; // Importing UserAge from a separate file
5
6  class App extends React.Component {
7    render() {
8      return (
9        <div>
10         <UserName />
11         <UserAge />
12       </div>
13     );
14   }
15 }
16
17 export default App;
```

Output:



3. How to add markup to JavaScript with JSX

Tasks:

1. Create a component that displays an unordered list () of 3 favorite fruits.

Code:

```
import './App.css';
function App() {
  return (
    <>
    <h1>Fruits</h1>
    <ul>
```

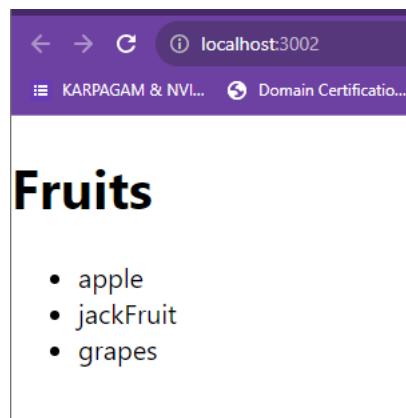


```

    <li>apple</li>
    <li>jackFruit</li>
    <li>grapes</li>
  </ul>
</>
);
}
export default App;

```

Output:



- Update the above component to display a picture () of each fruit next to its name. (Use hardcoded image URLs for now.)

Code:

```

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

function FruitList() {
  return (
    <ul>
      <li>Apple
        <br></br>
        
      </li>
      <li>Banana
        <br></br>

```

```

        
    </li>
    <li>Orange
        <br></br>
        
    </li>
</ul>
);
}

export default FruitList;

```

Output:



4. Create a component `WebsiteLink` that displays a hardcoded URL in an anchor (`<a>`) tag.

Code:

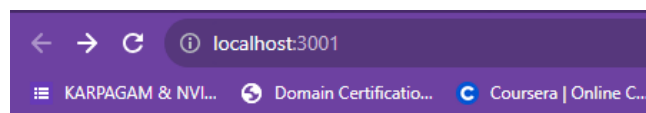
```
import './App.css';
import React from 'react';

class WebsiteLink extends React.Component {
  render() {
    // Hardcoded URL
    const url = 'https://legacy.reactjs.org/docs/getting-started.html';

    return (
      <>
        <h1>To start from React basics...</h1>
        <h2>
          <a href={url}>
            Visit Website
          </a>
        </h2>
      </>
    );
  }
}

export default WebsiteLink;
```

Output:



To start from React basics...

[Visit Website](https://legacy.reactjs.org/docs/getting-started.html)

4. Make a JSX component that mimics a simple blog post with a title, content, and author. (All hardcoded.)

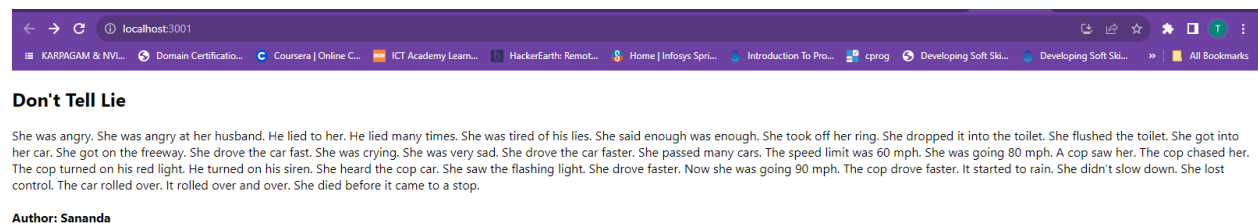
Code:

```
import './App.css';
import React from 'react';
function BlogPost() {
  const title = "Don't Tell Lie";
  const content = "She was angry. She was angry at her husband. He lied to her. He lied many times. She was tired of his lies. She said enough was enough. She took off her ring. She dropped it into the toilet. She flushed the toilet. She got into her car. She got on the freeway. She drove the car fast. She was crying. She was very sad. She drove the car faster. She passed many cars. The speed limit was 60 mph. She was going 80 mph. A cop saw her. The cop chased her. The cop turned on his red light. He turned on his siren. She heard the cop car. She saw the flashing light. She drove faster. Now she was going 90 mph. The cop drove faster. It started to rain. She didn't slow down. She lost control. The car rolled over. It rolled over and over. She died before it came to a stop.";
  const author = "Sananda";

  return (
    <div className="blog-post">
      <h2>{title}</h2>
      <div>
        <p>{content}</p>
      </div>
      <h4><p>Author: {author}</p></h4>
    </div>
  );
}
```

```
export default BlogPost;
```

Output:



5. Design a Footer component with hardcoded copyright information using JSX.

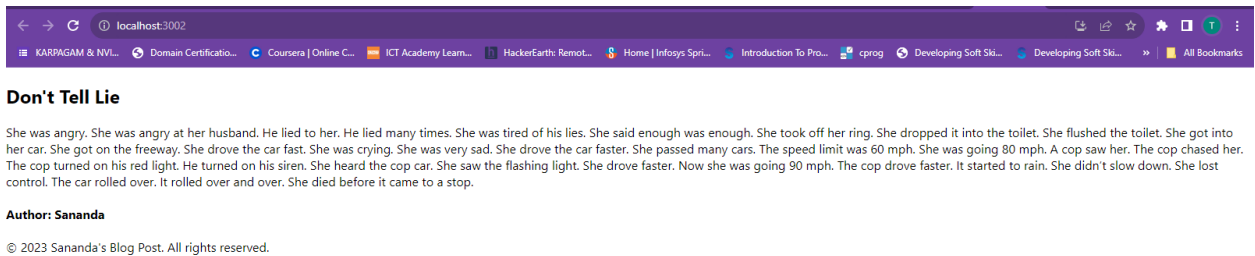
Code:

```
import React from 'react';

function Footer() {
  const title = "Don't Tell Lie";
  const content = "She was angry. She was angry at her husband. He lied to her. He lied many times. She was tired of his lies. She said enough was enough. She took off her ring. She dropped it into the toilet. She flushed the toilet. She got into her car. She got on the freeway. She drove the car fast. She was crying. She was very sad. She drove the car faster. She passed many cars. The speed limit was 60 mph. She was going 80 mph. A cop saw her. The cop chased her. The cop turned on his red light. He turned on his siren. She heard the cop car. She saw the flashing light. She drove faster. Now she was going 90 mph. The cop drove faster. It started to rain. She didn't slow down. She lost control. The car rolled over. It rolled over and over. She died before it came to a stop.";
  const author = "Sananda";
  const currentYear = new Date().getFullYear();
  const copyrightText = `© ${currentYear} Sananda's Blog Post. All rights reserved.`;
  return (
    <body>
      <div className="blog-post">
        <h2>{title}</h2>
        <div>
          <p>{content}</p>
        </div>
        <h4><p>Author: {author}</p></h4>
      </div>
      <footer className="footer">
        <div className="container">
          <p>{copyrightText}</p>
        </div>
      </footer>
    </body>
  );
}

export default Footer;
```

Output:



4.JavaScript in JSX with Curly Braces

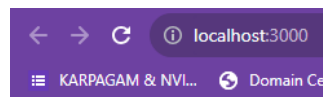
Tasks:

1. Display today's date in a component using the JavaScript Date object.

Code:

```
import React from 'react';
class DateComponent extends React.Component {
  render() {
    const today = new Date();
    const day = today.getDate();
    const month = today.toLocaleString('default', { month: 'long' });
    const year = today.getFullYear();
    const formattedDate = `${month} ${day}, ${year}`;
    return (
      <div>
        <h1>Today's Date</h1>
        <p>{formattedDate}</p>
      </div>
    );
  }
}
export default DateComponent;
```

Output:



Today's Date

September 26, 2023

2. Create a component that displays a random quote from a hardcoded list of quotes.

Code:

```
import React from 'react';
class RandomQuote extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      quotes: [
        "The only way to do great work is to love what you do. - Steve Jobs",
        "In three words I can sum up everything I've learned about life: it goes on. - Robert Frost",
        "Life is what happens when you're busy making other plans. - John Lennon",
        "Success is not final, failure is not fatal: It is the courage to continue that counts. - Winston Churchill",
        "The future belongs to those who believe in the beauty of their dreams. - Eleanor Roosevelt",
      ],
      randomQuote: ''
    };
  }
  componentDidMount() {
    this.getRandomQuote();
  }
  getRandomQuote = () => {
    const { quotes } = this.state;
    const randomIndex = Math.floor(Math.random() * quotes.length);
    const randomQuote = quotes[randomIndex];
    this.setState({ randomQuote });
  }
  render() {
    const { randomQuote } = this.state;
    return (
      <div>
        <h1>Random Quote</h1>
        <p>{randomQuote}</p>
        <button onClick={this.getRandomQuote}>Get Another Quote</button>
      </div>
    );
  }
}
export default RandomQuote;
```

Output:

Random Quote

Success is not final, failure is not fatal: It is the courage to continue that counts. - Winston Churchill

Get Another Quote

Random Quote

In three words I can sum up everything I've learned about life: it goes on. - Robert Frost

Get Another Quote

3. Write a component called MathResult that displays the result of a simple arithmetic operation (e.g., addition) of two hardcoded numbers.

```
import React from 'react';
class MathResult extends React.Component {
  render()
  {
    const num1 = 67,num2 = 89;
    const addedNum = num1+num2;
    return(
      <div>
        <h1>Adding Numbers</h1>
        <h3>{num1}+{num2}={addedNum}</h3>
      </div>
    );
  }
}
export default MathResult;
```


Output:

Adding Numbers

67+89=156

4. Create a component that displays the word count of a hardcoded paragraph.

Code:

```
import React from 'react';
class WordCount extends React.Component{
  render(){
    const Paragraph = 'The Solar System consists of the Sun Moon and Planets. It also consists of comets, meteoroids and asteroids. The Sun is the largest member of the Solar System. In order of distance from the Sun, the planets are Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune and Pluto; the dwarf planet. The Sun is at the centre of the Solar System and the planets, asteroids, comets and meteoroids revolve around it.';
    const words = Paragraph.split(/\s+/);
    const count = words.length;
    return(
      <>
        <h3>Paragraph Word Count</h3>
        <p>{Paragraph}</p>
        <h3>Total number of words is {count}</h3>
      </>
    );
  }
}
export default WordCount;
```

Output:

Paragraph Word Count

The Solar System consists of the Sun Moon and Planets. It also consists of comets, meteoroids and asteroids. The Sun is the largest member of the Solar System. In order of distance from the Sun, the planets are Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune and Pluto; the dwarf planet. The Sun is at the centre of the Solar System and the planets, asteroids, comets and meteoroids revolve around it.

Total number of words is 71

5. Create a component that calculates and displays the product of two hardcoded numbers.

Code:

```
import React from 'react';
class ProductCalculator extends React.Component {
  render() {
    const number1 = 5;
    const number2 = 10;
    const product = number1 * number2;
    return (
      <div>
        <h1>Product Calculator</h1>
        <p>
          Product of {number1} and {number2} is : {product}
        </p>
      </div>
    );
  }
}
export default ProductCalculator;
```

Output:

Product Calculator

Product of 5 and 10 is : 50

5. Passing Props to a Component

Tasks:

1. Create a Movie component that displays the title, year, and rating of a movie using props.

Code:

```
import React from 'react';
function Movie (props){
  const { title, year, rating } = props;
  return (
    <div>
      <h2>{title}</h2>
      <p>Year: {year}</p>
      <p>Rating: {rating}</p>
    </div>
  );
};
function App() {
  return (
    <div className="App">
      <h1>Movie Information</h1>
      <div>
        <Movie title="Inception" year="2010" rating="8.8" />
        <Movie title="The Shawshank Redemption" year="1994" rating="9.3" />
        <Movie title="The Dark Knight" year="2008" rating="9.0" />
      </div>
    </div>
  );
}
export default App;
```

Output:

Movie Information

Inception

Year: 2010

Rating: 8.8

The Shawshank Redemption

Year: 1994

Rating: 9.3

The Dark Knight

Year: 2008

Rating: 9.0

2. Update the Movie component to have a default prop for rating as “Not Rated”.

Code:

```
import React from 'react';
function Movie (props){
  const { title, year, rating } = props;
  return (
    <div>
      <h2>{title}</h2>
      <p>Year: {year}</p>
      <p>Rating: {rating}</p>
    </div>
  );
};
Movie.defaultProps = {
  rating: 'Not Rated',
};
function App() {
  return (
    <div className="App">
      <h1>Movie Information</h1>
      <div>
```

```

    <Movie title="Inception" year="2010" rating="8.8" />
    <Movie title="The Shawshank Redemption" year="1994" rating="9.3" />
    <Movie title="The Dark Knight" year="2008" rating="9.0" />
  </div>
</div>
);
}
export default App;

```

Output:

Movie Information

Inception

Year: 2010

Rating: 8.8

The Shawshank Redemption

Year: 1994

Rating: 9.3

The Dark Knight

Year: 2008

Rating: 9.0

3. Design a Button component that takes in a label prop and displays the label on the button.

Code:

```

import React from 'react';
function Button (props){
  return (
    <button>{props.label}</button>
  );
};
function App() {
  return (
    <div className="App">
      <h1>Button Example</h1>

```

```

        <Button label="Click Me" />
        <Button label="Submit" />
        <Button label="Cancel" />
      </div>
    );
  }
  export default App;

```

Output:

Button Example



4. Make a UserProfile component and pass an object containing user details as props and display them.

Code:

```

import React from 'react';
function UserProfile(props) {
  const { user } = props;
  return (
    <div>
      <h1>User Profile</h1>
      <p><h3>Name:</h3>{user.name}</p>
      <p><h3>Email: </h3>{user.email}</p>
      <p><h3>Age:</h3>{user.age}</p>
    </div>
  );
}
function App() {
  const user = {
    name: 'Sananda Srihari',
    email: '717821t140@kce.ac.in',
    age: 19,
  };
  return (
    <div className="App">
      <UserProfile user={user} />
    </div>
  );
}

```

```
);  
}  
export default App;
```

Output:

User Profile

Name:

Sananda Srihari

Email:

717821t140@kce.ac.in

Age:

19

5. Develop a Modal component that accepts and displays a title and some content passed as props.

Code:

```
import React, { useState } from 'react';  
function Modal(props) {  
  const { title, content, onClose } = props;  
  return (  
    <div>  
      <div>  
        <div>  
          <h2>{title}</h2>  
          <button onClick={onClose}>&times;</button>  
        </div>  
        <div>  
          <p>{content}</p>  
        </div>  
      </div>  
    </div>  
  );  
}  
function App() {  
  const [isModalOpen, setModalOpen] = useState(false);
```

```

const openModal = () => {
  setModalOpen(true);
};

const closeModal = () => {
  setModalOpen(false);
};

const modalContent = "Failure is the stepping stone to success!";

return (
  <div className="App">
    <h1>Hello User!</h1><button onClick={openModal}>Open title</button>
    {isModalOpen && (
      <Modal
        title="Modal Title : Success Journey"
        content={modalContent}
        onClose={closeModal}
      />
    )}
  </div>
);
}
export default App;

```

Output:

Hello User!

Open title

Hello User!

Open title

Modal Title : Success Journey

×

Failure is the stepping stone to success!

6. Conditional Rendering

Tasks:

1.Design a `UserStatus` component that displays “Online” or “Offline” based on a `isOnline` prop.

Code:

```
import React from 'react';
function UserStatus({ isOnline }) {
  const statusText = isOnline ? 'Online' : 'Offline';
  return (
    <div>
      <h3>Status: {statusText}</h3>
    </div>
  );
}
function App() {
  const isUserOnline = true; //else false so the userstatus will change into offline
  return (
    <div className="App">
      <h1>User Status</h1>
      <UserStatus isOnline={isUserOnline} />
    </div>
  );
}
export default App;
```

Output:

User Status

Status: Offline

User Status

Status: Online

2. Write a component AgeCheck that displays “Adult” or “Minor” based on an age prop.

Code:

```
import React from 'react';
function AgeCheck({ age }) {
  const statusText = age >= 18 ? 'Adult' : 'Minor';
  return (
    <div className="age-check">
      <p><h4>Status: </h4>{statusText}</p>
    </div>
  );
}
function App() {
  const userAge = 25; //when age value is given under 18 then status is changed
  into minor
  return (
    <div className="App">
      <h1>Age Check</h1>
      <AgeCheck age={userAge} />
    </div>
  );
}
export default App;
```

Output:

Age Check

Status:

Adult

Age Check

Status:

Minor

3. Create a Loading component that either displays “Loading...” or content based on a isLoading prop.

Code:

```
import React from 'react';
function Loading({ isLoading, content }) {
  return isLoading ? 'Loading...' : content;
}
function App() {
  const isLoading = false; //when value is true it displays Loading...
  const exampleContent = <div><h2>Content Loaded</h2><p>This is Sananda srihari.</p></div>;
  return (
    <div className="App">
      <h1>Loading Example</h1>
      <Loading isLoading={isLoading} content={exampleContent} />
    </div>
  );
}
export default App;
```

Output:

Loading Example

Content Loaded

This is Sananda srihari.

Loading Example

Loading...

4. Make a Notification component that conditionally displays a message if a message prop is provided.

Code:

```
import React from 'react';
function Notification({ message }) {
  return message ? <div className="notification">{message}</div> : null;
}
function App() {
  const notificationMessage = 'You got a notification from mail';
  return (
    <div className="App">
      <h1>Notification</h1>
      <Notification message={notificationMessage} />
    </div>
  );
}
export default App;
```

Output:

Notification

You got a notification from mail

4. Design a Feedback component that displays feedback in either green (positive) or red (negative) based on a type prop.

Code:

```
import React from 'react';
function Feedback({ type, message }) {
  const feedbackStyle = {
    color: type === 'positive' ? 'green' : 'red',
  };
  return (
    <div style={feedbackStyle}>
      {message}
    </div>
  );
}
```

```
function App() {
  const positiveFeedback = {
    type: 'positive',
    message: 'Great job! Your feedback is positive.',
  };
  const negativeFeedback = {
    type: 'negative',
    message: 'Oops! Your feedback is negative.',
  };
  return (
    <div className="App">
      <h1>Feedback Example</h1>
      <Feedback {...positiveFeedback} />
      <Feedback {...negativeFeedback} />
    </div>
  );
}
export default App;
```

Output:

Feedback Example

Great job! Your feedback is positive.
Oops! Your feedback is negative.

7.Rendering Lists

Tasks:

1. Write a component that takes an array of names as a prop and displays them in a list.

Code:

```
import React from 'react';
function NameList({ names }) {
  return (
    <ul>
      {names.map((name, index) => (
        <li key={index}>{name}</li>
      ))}
    </ul>
  );
}
```

```

    ))}
  </ul>
);
}
function App() {
  const names = ['Sana', 'Jaga', 'kishore', 'rameez', 'bala'];
  return (
    <div>
      <h1>Name List</h1>
      <NameList names={names} />
    </div>
  );
}
export default App;

```

Output:

Name List

- Sana
- Jaga
- kishore
- rameez
- bala

2. Create a TodoList component that displays a list of tasks and marks the completed ones.

Code:

```

import React from 'react';
function TodoList({ tasks }) {
  return (
    <ul>
      {tasks.map((task, index) => (
        <li key={index} className={task.completed ? 'completed' : ''}>
          {task.text}
        </li>
      ))}
    </ul>
  );
}

```

```
function App() {
  const tasks = [
    { text: 'Buy groceries', completed: false },
    { text: 'Read a book', completed: true },
    { text: 'Write code', completed: false },
  ];
  return (
    <div>
      <h1>To do List</h1>
      <ToDoList tasks={tasks} />
    </div>
  );
}
export default App;
```

Output:

To do List

- Buy groceries
- Read a book
- Write code

3.Design a ProductList component that only displays products with a price less than \$10 using the filter() method.

Code:

```
import React from 'react';
function ProductList({ products }) {
  const affordableProducts = products.filter((product) => product.price < 10);
  return (
    <div>
      <h2>Affordable Products (Less than Rs.10)</h2>
      <ul>
        {affordableProducts.map((product, index) => (
          <li key={index}>
            {product.name} - ${product.price}
          </li>
        ))}
      </ul>
    </div>
```

```

    );
  }
  function App() {
    const products = [
      { name: 'Product A', price: 8},
      { name: 'Product B', price: 12},
      { name: 'Product C', price: 5},
      { name: 'Product D', price: 7},
    ];
    return (
      <div>
        <h1>Product List</h1>
        <ProductList products={products} />
      </div>
    );
  }
  export default App;

```

Output:

Product List

Affordable Products (Less than Rs.10)

- Product A - \$8
- Product C - \$5
- Product D - \$7

4. Make a UserList component that takes an array of user objects and displays their names and emails.

Code:

```

import React from 'react';
function UserList({ users }) {
  return (
    <div>
      <h2>User List</h2>
      <ul>
        {users.map((user, index) => (
          <li key={index}>

```



```

        <strong>Name:</strong> {user.name}, <strong>Email:</strong>
{user.email}
      </li>
    )))
  </ul>
</div>
);
}
function App() {
  const users = [
    { name: 'Sananda', email: 'Sana@01.com' },
    { name: 'Suba', email: 'Suba@02.com' },
    { name: 'Jaga', email: 'Jaga@03.com' },
  ];
  return (
    <div>
      <h1>User List & their details</h1>
      <UserList users={users} />
    </div>
  );
}
export default App;

```

Output:

User List & their details

User List

- **Name:** Sananda, **Email:** Sana@01.com
- **Name:** Suba, **Email:** Suba@02.com
- **Name:** Jaga, **Email:** Jaga@03.com

5. Create a ShoppingCart component that displays a list of items and their prices. Ensure each item has a unique key.

Code:

```

import React from 'react';
function ShoppingCart({ items }) {
  return (
    <div>
      <ul>

```

```

        {items.map((item, index) => (
          <li key={index}>
            <strong>{item.name}</strong> - ${item.price}
          </li>
        ))}
      </ul>
    </div>
  );
}
function App() {
  const items = [
    { name: 'Item A', price: 9.99 },
    { name: 'Item B', price: 14.95 },
    { name: 'Item C', price: 5.49 },
  ];
  return (
    <div>
      <h1>Shopping Cart</h1>
      <ShoppingCart items={items} />
    </div>
  );
}
export default App;

```

Output:

Shopping Cart

- **Item A** - \$9.99
- **Item B** - \$14.95
- **Item C** - \$5.49

8. Keeping Components Pure

Tasks:

1. Convert an impure component that uses `Math.random()` within the render phase to a pure one.

Code:

```
import React, { Component } from 'react';
class RandomNumberDisplay extends Component {
  state = {
    randomNumber: Math.random(),
  };
  render() {
    return (
      <div>
        <h1>Random Number</h1>
        <p>{this.state.randomNumber}</p>
      </div>
    );
  }
}
export default RandomNumberDisplay;
```

Output:

Random Number

0.39874745733290284

2. Create a pure component `Clock` that displays the current time and updates every second without causing side-effects during the render phase.

Code:

```
import React, { Component } from 'react';
class Clock extends Component {
  constructor(props) {
    super(props);
    this.state = {
      time: new Date().toLocaleTimeString(),
    };
  }
}
```

```

    };
  }
  componentDidMount() {
    this.timerID = setInterval(() => this.tick(), 1000);
  }
  componentWillUnmount() {
    clearInterval(this.timerID);
  }
  tick() {
    this.setState({
      time: new Date().toLocaleTimeString(),
    });
  }
  render() {
    return (
      <div>
        <h1>Current Time:</h1>
        <p>{this.state.time}</p>
      </div>
    );
  }
}
export default Clock;

```

Output:

Current Time:

12:25:13

3. Use Strict Mode in an existing application and identify any warnings in the console.

Code:

```

import React from 'react';
function MyComponent() {
  <h1>hello</h1>
}
function App() {
  return (
    <div>

```

```

    <h1>My Page</h1>
    <React.StrictMode>
      <MyComponent />
    </React.StrictMode>
  </div>
);
}
export default App;

```

Output:

My Page

4. Convert a class-based component with side effects in its lifecycle methods to a pure functional component using hooks.

Code:

```

import React, { useState, useEffect } from 'react';
function Timer() {
  const [seconds, setSeconds] = useState(0);
  useEffect(() => {
    const timerID = setInterval(() => {
      setSeconds((prevSeconds) => prevSeconds + 1);
    }, 1000);
    return () => {
      clearInterval(timerID);
    };
  }, []);
  return (
    <div>
      <h1>Timer</h1>
      <p>Seconds: {seconds}</p>
    </div>
  );
}
export default Timer;

```

Output:

Timer

Seconds: 20

5. Make a pure ProfilePic component that takes a user ID as a prop and fetches the user's profile picture URL from an array without side-effects during rendering.

Code:

```
import React from 'react';
const userProfiles = [
  { id: 1, profilePicUrl: 'https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcQ9qMp3RM66v5flm1qdQm-
xI8qajl0USrQT2A&usqp=CAU' },
  { id: 2, profilePicUrl: 'https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcQTU0sKK6aufgDha4BF_Yf-
2DFmoBPlzYwyMw&usqp=CAU' },
  { id: 3, profilePicUrl: 'https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcTJRv5dEZTaidyHavbXn3ZKklLufAj_NBD_ng&usqp=CAU
' },
];
function ProfilePic({ userId }) {
  const userProfile = userProfiles.find((profile) => profile.id === userId);
  return (
    <div>
      {userProfile ? (
        <>
          <h2>User Profile Picture</h2>
          <img src={userProfile.profilePicUrl} alt="Profile Pic" />
        </>
      ) : (
        <div>User not found</div>
      )}
    </div>
  );
}
export default ProfilePic;
```

Output:

User not found