

# ReactJS Practice Question Set 7

## Questions:

1. Create a React component that fetches weather data from an API endpoint using `useEffect` hook and displays the current temperature, humidity, and wind speed on the screen using the `useState` hook. Add a button which toggles between Celsius and Fahrenheit units for the temperature.

```
const fakeFetch = (url) => {  
  return new Promise((resolve, reject) => {  
    setTimeout(() => {  
      if (url === 'https://example.com/api/weather') {  
        resolve({  
          status: 200,  
          message: 'Success',  
          data: {  
            temperature: 21,  
            humidity: 50,  
            windSpeed: 10,  
          },  
        })  
      } else {  
        reject({  
          status: 404,  
          message: 'Weather data not found.',  
        })  
      }  
    }, 2000)  
  })  
}
```

[COPY](#)

## Expected Output

# Weather

Temperature: 21°C

Humidity: 50%

Wind Speed: 10 km/h

Switch to Fahrenheit

On clicking the button

# Weather

Temperature: 70°F

Humidity: 50%

Wind Speed: 10 km/h

Switch to Celsius

2. Create a React component that fetches user data from an API endpoint using `useEffect` hook and displays the user's name, email, and phone number on the screen using the `useState` hook. Add a button which toggles the display of the user's address (street, suite, city, zipcode).

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === 'https://example.com/api/user') {
        resolve({
          status: 200,
          message: 'Success',
          data: {
            name: 'John Doe',
            email: 'john.doe@example.com',
            phone: '+1 555-555-5555',
            address: {
              street: '123 Main St',
              suite: 'Suite 456',
              city: 'Anytown',
              zipcode: '12345',
            },
          },
        })
      }
    }, 1000)
  })
}
```

```
    })  
  } else {  
    reject({  
      status: 404,  
      message: 'User not found.',  
    })  
  }  
}, 2000)  
})  
}
```

[COPY](#)

## Expected Output

# User

Name: John Doe

Email: john.doe@example.com

Phone: +1 555-555-5555

Hide Address

*123 Main St*

*Suite 456*

*Anytown*

*12345*

On clicking the button

# User

Name: John Doe

Email: john.doe@example.com

Phone: +1 555-555-5555

Hide Address

*123 Main St*

*Suite 456*

*Anytown*

*12345*

3. Create a React component that fetches a list of movies from an API endpoint using `useEffect` hook and displays the title, year, and rating of each movie on the screen using the `useState` hook. Add a dropdown which filters the movies by year. You can keep 5 dropdown values - 2005 to 2010.


```
const fakeFetch = (url) => {  
  return new Promise((resolve, reject) => {
```

```
setTimeout(() => {
  if (url === 'https://example.com/api/movies') {
    resolve({
      status: 200,
      message: 'Success',
      data: [
        { title: 'The Dark Knight', year: 2008, rating: 9.0 },
        { title: 'Inception', year: 2009, rating: 8.8 },
        { title: 'Interstellar', year: 2010, rating: 8.6 },
        { title: 'Tenet', year: 2009, rating: 7.5 },
        { title: 'Real Steal', year: 2007, rating: 7.5 },
      ],
    })
  } else {
    reject({
      status: 404,
      message: 'Movies list not found.',
    })
  }
}, 2000)
})
}
```

COPY

## Expected Output

# Movies

Filter by Year:  

- Name: The Dark Knight

Year:2008

Ratings: 9

- Name: Inception

Year:2009

Ratings: 8.8

- Name: Interstellar

Year:2010

Ratings: 8.6



- Name: Tenet

Year:2009

Ratings: 7.5

On typing the year, it will filter

# Movies

Filter by Year: 2009 ▼

- Name: Inception

Year: 2009

Ratings: 8.8

- Name: Tenet

Year: 2009

Ratings: 7.5

4. Create a React component that fetches a list of users from an API endpoint using `useEffect` hook and displays the name, email, and website of each user on the screen using the `useState` hook. Add a dropdown which filters the users by company name.

```
const fakeFetch = (url) => {  
  return new Promise((resolve, reject) => {  
    setTimeout(() => {  
      if (url === 'https://example.com/api/users') {  
        resolve({  
          status: 200,  
          message: 'Success',  
          data: [  
            {
```

```

        name: 'John Doe',
        email: 'john@example.com',
        website: 'example.com',
        company: 'ABC Inc',
      },
      {
        name: 'Jane Doe',
        email: 'jane@example.com',
        website: 'example.com',
        company: 'XYZ Corp',
      },
      {
        name: 'Bob Smith',
        email: 'bob@example.com',
        website: 'example.com',
        company: 'ABC Inc',
      },
      {
        name: 'Alice Brown',
        email: 'alice@example.com',
        website: 'example.com',
        company: 'ACME Corp',
      },
      {
        name: 'Charlie Green',
        email: 'charlie@example.com',
        website: 'example.com',
        company: 'XYZ Corp',
      },
    ],
  })
} else {
  reject({
    status: 404,
    message: 'Users list not found.',
  })
}
}, 2000)
})
}

```

COPY

## Expected Output

# Users

Filter by Company: All companies ▼

- John Doe

john@example.com

example.com

ABC Inc

- Jane Doe

jane@example.com

example.com

XYZ Corp

- Bob Smith

bob@example.com

example.com


ABC Inc

- Alice Brown

alice@example.com

On typing the name, it will filter out

# Users

Filter by Company:  

- Jane Doe

jane@example.com

example.com

XYZ Corp

- Charlie Green

charlie@example.com

example.com

XYZ Corp

5. Create a component that displays a random quote from an API using the `useEffect` and `useState` hooks. The component should fetch a new quote when the user clicks a button.

```
const fakeFetch = () => {
  const quotes = [
    {
      content: 'Be yourself; everyone else is already taken.',
      author: 'Oscar Wilde',
    },
    {
      content:
        "Two things are infinite: the universe and human stupidity; and I'm not sure about the",
      author: 'Albert Einstein',
    },
    {
      content: 'So many books, so little time.',
      author: 'Frank Zappa',
    },
    {
      content: 'A room without books is like a body without a soul.',
      author: 'Marcus Tullius Cicero',
    },
    {
      content:
        "In three words I can sum up everything I've learned about life: it goes on.",
      author: 'Robert Frost',
    },
  ]

  return new Promise((resolve) => {
    setTimeout(() => {
      const randomQuote = quotes[Math.floor(Math.random() * quotes.length)]
      resolve(randomQuote)
    }, 1000)
  })
}
```

COPY

## Expected Output

"Two things are infinite: the universe and human stupidity; and I'm not sure about the universe."

- Albert Einstein

New Quote

Clicking the button will generate a new quote

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

- Robert Frost

New Quote

6. Create a React component that fetches a list of movies from an API endpoint using `useEffect` hook and displays the title, year, and genre of each movie on the screen using the `useState` hook. Add a dropdown which filters the movies by genre.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === 'https://example.com/api/movies') {
        resolve({
          status: 200,
          message: 'Success',
          data: [
            {
              title: 'The Godfather',
              year: 1972,
              genre: 'Crime',
            },
            {
              title: 'The Shawshank Redemption',
              year: 1994,
              genre: 'Drama',
            },
            {
              title: 'The Dark Knight',
              year: 2008,
              genre: 'Action',
            },
            {
              title: 'Forrest Gump',
              year: 1994,
              genre: 'Comedy',
            },
            {
              title: 'The Matrix',
              year: 1999,
              genre: 'Science Fiction',
            },
            {
              title: 'Jurassic Park',
              year: 1993,
              genre: 'Science Fiction',
            },
            {
              title: 'Star Wars: Episode IV - A New Hope',
              year: 1977,
              genre: 'Science Fiction',
            },
            {
              title: 'The Terminator',
              year: 1984,
              genre: 'Action',
            },
          ],
        })
      }
    }, 1000)
  })
}
```




```
    },
    {
      title: 'Die Hard',
      year: 1988,
      genre: 'Action',
    },
    {
      title: 'Pulp Fiction',
      year: 1994,
      genre: 'Crime',
    },
  ],
  })
} else {
  reject({
    status: 404,
    message: 'Movies list not found.',
  })
}
}, 2000)
})
}
```

COPY

## Expected Output

# Movies

Filter by Genre:  

- The Godfather

1972

Crime

- The Shawshank Redemption

1994

Drama

- The Dark Knight

2008

Action

- Forrest Gump

1994

Comedy

- The Matrix

1999

---

On typing the genre, it will filter out

---

# Movies

Filter by Genre:

- The Godfather

1972

Crime

- Pulp Fiction

1994

Crime

7. Create a React component that fetches a list of products from an e-commerce API endpoint using `useEffect` hook and displays the product name, description, price, and quantity on the screen using the `useState` hook. Add a button which allows the user to sort the products by price (lowest to highest).

```
const fakeFetch = (url) => {  
  return new Promise((resolve, reject) => {  
    setTimeout(() => {  
      if (url === 'https://example.com/api/products') {  
        resolve({  
          status: 200,  
          message: 'Success',  
        })  
      }  
    }, 1000)  
  })  
}
```

```

data: {
  products: [
    {
      name: 'Product 1',
      description: 'This is the first product',
      price: 25.99,
      quantity: 10,
    },
    {
      name: 'Product 2',
      description: 'This is the second product',
      price: 19.99,
      quantity: 15,
    },
    {
      name: 'Product 3',
      description: 'This is the third product',
      price: 35.5,
      quantity: 5,
    },
    {
      name: 'Product 4',
      description: 'This is the fourth product',
      price: 49.99,
      quantity: 20,
    },
  ],
}
}))
} else {
  reject({
    status: 404,
    message: 'Product list not found.',
  })
}
}, 2000)
})
}

```

[COPY](#)

## Expected Output

# Products

Sort by Price

- **Product 1**

This is the first product

\$25.99

Quantity: 10

- **Product 2**

This is the second product

\$19.99

Quantity: 15

- **Product 3**

This is the third product

\$35.5

Quantity: 5

On clicking the button it will sort it out by price

# Products

Sort by Price

- **Product 2**

This is the second product

\$19.99

Quantity: 15

- **Product 1**

This is the first product

\$25.99

Quantity: 10



# • Product 3

This is the third product

\$35.5

Quantity: 5

8. Adding on to the previous question, There should be three buttons for this purpose: "Low to High", "High to Low", and "Reset". When the user clicks on "Low to High", the products should be sorted by price in ascending order. When the user clicks on "High to Low", the products should be sorted by price in descending order. When the user clicks on "Reset", the products should be displayed in their original order.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === 'https://example.com/api/products') {
        resolve({
          status: 200,
          message: 'Success',
          data: {
            products: [
              {
                name: 'Product 1',
                description: 'This is the first product',
                price: 25.99,
                quantity: 10,
              },
              {
                name: 'Product 2',
                description: 'This is the second product',
                price: 19.99,
                quantity: 15,
              },
              {
                name: 'Product 3',
                description: 'This is the third product',
                price: 35.5,
                quantity: 5,
              },
              {
                name: 'Product 4',
                description: 'This is the fourth product',
                price: 49.99,
                quantity: 20,
              }
            ]
          }
        })
      }
    }, 1000)
  })
}
```

```
        },  
      ],  
    },  
  })  
} else {  
  reject({  
    status: 404,  
    message: 'Product list not found.',  
  })  
}  
}, 2000)  
})  
}
```

[COPY](#)

## Expected Output

Low to High

High to Low

Reset

Product 1

25.99

10

Product 2

19.99

15

Product 3

35.5

5

Product 4

49.99

20

On click Low to High

Low to High

High to Low

Reset

Product 2

19.99

15

Product 1

25.99

10

Product 3

35.5

5

Product 4

49.99

20

On click High to Low

Low to High

High to Low

Reset

Product 4

49.99

20

Product 3

35.5

5

Product 1

25.99

10

Product 2

19.99

15

On click the Reset button

Low to High

High to Low

Reset

Product 2

19.99

15

Product 1

25.99

10

Product 3

35.5

5

Product 4

49.99

20

9. Create a React component that uses the `useEffect` hook to fetch the product data from the API endpoint using the `fakeFetch` function provided below. The component should use the `useState` hook to store the fetched data and a second state variable to store the sorted data. The sorted data should be sorted in descending order by rating.

```
const fakeFetch = (url) => {  
  return new Promise((resolve, reject) => {  
    setTimeout(() => {  
      if (url === 'https://example.com/api/products') {  
        resolve({  
          status: 200,  
          message: 'Success',  
          data: {  
            products: [  
              { name: 'Color Pencils', price: 50, quantity: 40, rating: 4.5 },  
              { name: 'Sketchpens', price: 110, quantity: 20, rating: 3.8 },  
              { name: 'Eraser', price: 20, quantity: 20, rating: 4.2 },  
              { name: 'Sharpener', price: 22, quantity: 30, rating: 4.7 },  
            ],  
          },  
        })  
      } else {
```

```
    reject({
      status: 404,
      message: 'Product list not found.',
    })
  }
}, 2000)
})
}
```

[COPY](#)

## Expected Output

### Products

Sharpener

Rating: 4.7

Price: \$22

Quantity: 30

Color Pencils

Rating: 4.5

Price: \$50

Quantity: 40

Eraser

Rating: 4.2

Price: \$20

Quantity: 20

Sketchpens

Rating: 3.8

10. Adding on to the previous question, Add a search bar to the component that allows users to filter the products by name. The search bar should update the list of displayed products in real-time

as the user types. The search functionality should be case-insensitive.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === 'https://example.com/api/products') {
        resolve({
          status: 200,
          message: 'Success',
          data: {
            products: [
              { name: 'Color Pencils', price: 50, quantity: 40, rating: 4.5 },
              { name: 'Sketchpens', price: 110, quantity: 20, rating: 3.8 },
              { name: 'Eraser', price: 20, quantity: 20, rating: 4.2 },
              { name: 'Sharpener', price: 22, quantity: 30, rating: 4.7 },
            ],
          },
        })
      } else {
        reject({
          status: 404,
          message: 'Product list not found.',
        })
      }
    }, 2000)
  })
}
```

[COPY](#)

## Expected Output

### Products

Color Pencils

Rating: 4.5

Price: \$50

Quantity: 40