Conditional Rendering with &&

```
const Footer = ()=>
{
    const hour = new Date().getHours();
    const openHour = 10;
    const closeHour = 22;
    const isOpen = hour >= openHour && hour <= closeHour;

    return (
    <footer className="footer">
        { isOpen && (
        <div className='order'>
            <p>We are open until {closeHour}:00. Come visit us or order
online. </p>
            <button className="btn">Order</button>
        </div>)}
    </footer>)
}

const Menu = ()=>
{
    return (
    <main className='menu'>
        <h2>Our Menu</h2>
        { pizzaData && (<ul className='pizzas'>{ pizzaData.map((pizza,
idx)=> (<Pizza {...pizza} key={idx} /> ) ) }</ul>) }
    </main>)
}
```

Conditional Rendering with Ternaries

```
const Menu = () =>
{
    return (
    <main className='menu'>
        <h2>Our Menu</h2>
        {pizzaData ? (
            <ul className='pizzas'>
                {pizzaData.map((pizza, idx) => ( <Pizza {...pizza}
key={idx} /> ))}
            </ul>
        ) : (
          <p>Loading...</p>
        )}
    </main>
    )
}
```

```
const Footer = () =>
{
    const hour = new Date().getHours();
    const openHour = 10;
    const closeHour = 22;
    const isOpen = hour >= openHour && hour <= closeHour;
    return (
      <footer className="footer">
        {isOpen ? (
          <div className='order'>
            <p>We are open until {closeHour}:00. Come visit us or order
online. </p>
            <button className="btn">Order</button>
          </div>
        ) : (
          <p>Sorry, we are currently closed.</p>
        )}
      </footer>
    )
}
```

Conditional Rendering with Multiple Returns

```
const Footer = () =>
{
    const hour = new Date().getHours();
    const openHour = 10;
    const closeHour = 22;
    const isOpen = hour >= openHour && hour <= closeHour;

    if (!isOpen)
    {
      return (<p>Sorry, we are currently closed.</p>);
    }

    return (
      <footer className="footer">
        <div className='order'>
          <p>We are open until {closeHour}:00. Come visit us or order
online. </p>
          <button className="btn">Order</button>
        </div>
      </footer>
    )
}
```

Extracting JSX into a New Component

```
const Order = ({closeHour})=>
{
    return (
    <div className='order'>
        <p>We are open until {closeHour}:00. Come visit us or order online.
</p>
        <button className="btn">Order</button>
    </div>)
}

const Timings = ({ openHour, closeHour })=>
{
    return (<p> We are happy to welcome you between {openHour}:00 and
{closeHour}:00 </p>)
}


const Footer = ()=>
{
    const hour = new Date().getHours();
    const openHour = 10;
    const closeHour = 22;
    const isOpen = hour >= openHour && hour <= closeHour;

    return (
    <footer className="footer">
        { isOpen ? (<Order closeHour={closeHour}/> ): (<Timings
openHour={openHour} closeHour={closeHour}/> ) }
    </footer>)
}
```

When a JSX in a component becomes too big, extract that JSX into it’s own component. If a JSX depends upon some data that belongs to the parent component, we can pass that data as props.

Setting Classes and Text Conditionally

```
const Pizza = ({ name, photoName, price, ingredients, soldOut })=>
{
    return (
    <li className={soldOut ? "pizza sold-out" : "pizza"}>
        <img src={photoName} alt={name} />
        <div>
            <h3>{name}</h3>
            <p>{ingredients}</p>
            <span>{ soldOut? "SOLD OUT" : price }</span>
        </div>
    </li>);
}
```