```jsx
import React,{useState} from "react";

export default function Mode() {

  const [mystyle, setMystyle] = useState({
    color : 'black',
    backgroundColor : 'white',

      })

  const [btntext, setbtntext] = useState("Enable light Mode")

  const toggleButton = ()=>
  {
    if(mystyle.color === "white"){
      setMystyle(
        {
          color : 'black',
          backgroundColor : 'white',

      }
      )
      setbtntext("Enable dark mode")
    }
    else{
      setMystyle(
        {
          color : 'white',
          backgroundColor : 'black',
          border : '2px solid white'
      }
      )
      setbtntext("Enable light mode")

    }
  }


  return (
    <>
      <div className="container border border-2 border my-2 rounded rounded-3 py-
3" style={mystyle}>
        <h1 className=" text-center">Enable Mode</h1>
        <div className="accordion my-3" id="accordionExample">
          <div className="accordion-item">
```

```
    <h2 className="accordion-header">
      <button
        className="accordion-button"
        type="button"
        data-bs-toggle="collapse"
        data-bs-target="#collapseOne"
        aria-expanded="true"
        aria-controls="collapseOne"
        style={mystyle}
      >
        Accordion Item #1
      </button>
    </h2>
    <div
      id="collapseOne"
      className="accordion-collapse collapse show"
      data-bs-parent="#accordionExample"
    >
      <div className="accordion-body" style={mystyle}>
        <strong>This is the first item's accordion body.</strong> It is
        shown by default, until the collapse plugin adds the appropriate
        classes that we use to style each element. These classes control
        the overall appearance, as well as the showing and hiding via
        CSS transitions. You can modify any of this with custom CSS or
        overriding our default variables. It's also worth noting that
        just about any HTML can go within the{" "}
        <code>.accordion-body</code>, though the transition does limit
        overflow.
      </div>
    </div>
  </div>
  <div className="accordion-item">
    <h2 className="accordion-header">
      <button
        className="accordion-button collapsed"
        type="button"
        data-bs-toggle="collapse"
        data-bs-target="#collapseTwo"
        aria-expanded="false"
        aria-controls="collapseTwo"
        style={mystyle}
      >
        Accordion Item #2
      </button>
    </h2>
```

```
      <div
        id="collapseTwo"
        className="accordion-collapse collapse"
        data-bs-parent="#accordionExample"
      >
        <div className="accordion-body" style={mystyle}>
          <strong>This is the second item's accordion body.</strong> It is
          hidden by default, until the collapse plugin adds the
          appropriate classes that we use to style each element. These
          classes control the overall appearance, as well as the showing
          and hiding via CSS transitions. You can modify any of this with
          custom CSS or overriding our default variables. It's also worth
          noting that just about any HTML can go within the{" "}
          <code>.accordion-body</code>, though the transition does limit
          overflow.
        </div>
      </div>
    </div>
    <div className="accordion-item">
      <h2 className="accordion-header">
        <button
          className="accordion-button collapsed"
          type="button"
          data-bs-toggle="collapse"
          data-bs-target="#collapseThree"
          aria-expanded="false"
          aria-controls="collapseThree"
          style={mystyle}
        >
          Accordion Item #3
        </button>
      </h2>
      <div
        id="collapseThree"
        className="accordion-collapse collapse"
        data-bs-parent="#accordionExample"
      >
        <div className="accordion-body" style={mystyle}>
          <strong>This is the third item's accordion body.</strong> It is
          hidden by default, until the collapse plugin adds the
          appropriate classes that we use to style each element. These
          classes control the overall appearance, as well as the showing
          and hiding via CSS transitions. You can modify any of this with
          custom CSS or overriding our default variables. It's also worth
          noting that just about any HTML can go within the{" "}
```

```
              <code>.accordion-body</code>, though the transition does limit
              overflow.
            </div>
          </div>
        </div>
          <button type="submit" onClick={toggleButton} className="btn btn-
primary w-100 my-3">{btntext}</button>
        </div>
      </div>
    </>
  );
}
```