

2)

a => <https://codesandbox.io/s/learning-react-star-rating-3-tpmr9?file=/src/StarRating.js>

```
<>
{createArray(totalStars).map((n, i) => (
  <Star key={i} selected={selectedStars > i} />
))}
<p>
```

Se usa las props, que proviene del useState()

3)

```
JS App.js x
1 import React from "react";
2 import StartRating from "../StarRating";
3
4 export default function App() {
5   return <StartRating totalStars={10} />;
6 }
```

Browser Tests

★★★★★★★★★★

```
JS App.js x
1 import React from "react";
2 import StartRating from "../StarRating";
3
4 export default function App() {
5   return <StartRating />;
6 }
```

Browser Tests

★★★★★

3 of 5 stars

1) En la primer imagen se cambia el valor por defecto del número de estrellas a 10 y en el segundo queda por defecto que es 5.

```
export default function StarRating({ totalStars = 5 }) {  
  return createArray(totalStars).map((n, i) => <Star key={i} />);  
}
```

2

3

```
export default function StarRating({ totalStars = 5 }) {  
  const [selectedStars] = useState(3);  
  return (  
    <>  
      {createArray(totalStars).map((n, i) => (  
        <Star key={i} selected={selectedStars > i} />  
      ))}  
      <p>  
        {selectedStars} of {totalStars} stars  
      </p>  
    </>  
  );  
}
```

2

3

- 2) En la primer imagen , vemos la ausencia del hookk useState que cambia la property de selectedStars.
- 3) En la segunda, el return devuelve mas de un elemento, por ello usa los fragments para agrupar (<></>), Además se pasa como property el valor del useState al componente Star.