

S06 Book Checkout

BOOK CHECKOUT COMPONENT OVERVIEW

Book Checkout Component

- User will be able to see book info, checkout info and latest reviews

Desktop

Mobile

Book info Module:

Book Image

Book Title

Book Author

Book Description

Overall Book Star Rating

Checkout info Module:

The screenshot displays a user interface for managing book availability and reviews. At the top left, a blue box labeled "Books left that a user can checkout" shows "0/5 books checked out". Below it, a section titled "Book availability" indicates "Available" status with "10 copies" and "10 available". A "Sign in" button is present. A note states: "This number can change until placing order has been complete." To the right, a blue box labeled "Total books that are still available to be checked out" shows "10 available". Below these sections, a blue box labeled "Total copies of this book" displays a star rating of "★★★★★". The main content area shows a review by "Luv, Zofia" dated "September 5, 2022", rating "★★★★★", and the text "First book is pretty good book overall". A "Reach all reviews" button is at the bottom left, and a video camera icon is on the right.

Latest review Module:

The screenshot shows a review for the book "JavaScript Cookbook" by "Zofia Luv". The book cover is displayed on the left. A blue box labeled "Latest reviews for this specific book" contains a review by "Luv, Zofia" dated "September 5, 2022", rating "★★★★★", and the text "First book is pretty good book overall". A "Reach all reviews" button is below it. Another blue box labeled "Email, date and description" shows the same review details. To the right, a section titled "Review rating" displays the star rating "★★★★★". Above this, a "Sign in" button and a note about order completion are visible. The main content area also includes a "Available" status with "10 copies" and "10 available", and a "Sign in" button.

Book Checkout Component

- Four new TypeScript components

Desktop

BookCheckoutPage.tsx

CheckoutAndReviewBox.tsx

LatestReviews.tsx

StarsReview.tsx

JavaScript Cookbook
Luv, Zofia
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin risus tortor, condimentum eget sapien ac, dapibus varius ligula. Maecenas justo erat, semper sed nunc vel, vulputate eleifend du. Integer id ipsum vitae nisl malesuada feugiat. Proin sit amet quam laoreet, feugiat mi vitae, vestibulum du. Aliquam erat volutpat. Etiam hendrerit erat nec mi auctor elementum. Curabitur vestibulum lectus a ante tempor tincidunt et sed orci. Proin maximus tortor in risus auctor efficitur. Phasellus quam mauris, laoreet et feugiat ac, imperdiet at quam. Nullam sollicitudin nec diam vel finibus.

★★★★★

Latest Reviews:
exampleUser@email.com
September 5, 2022
First book is pretty good book overall
★★★★★

Reach all reviews

0/5 books checked out

Available
10 copies 10 available

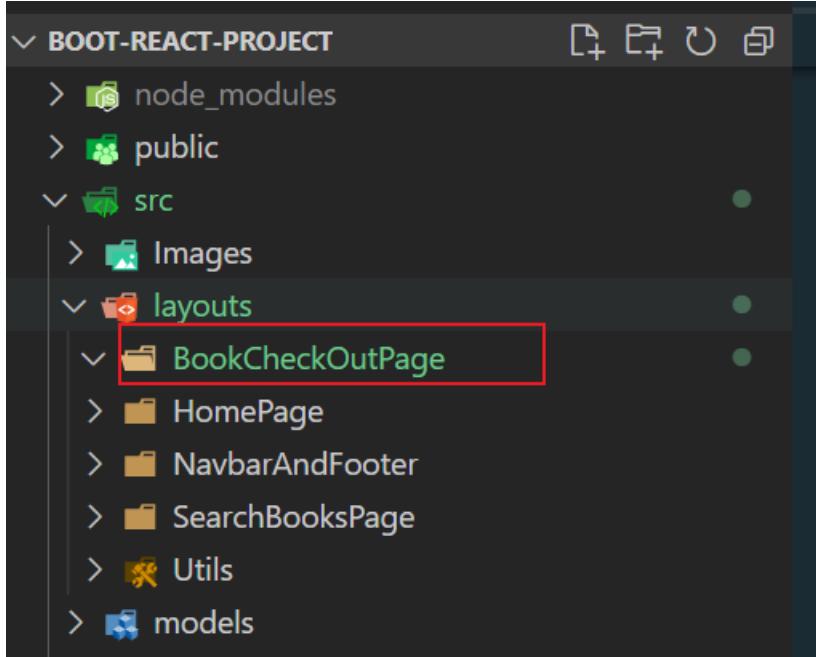
Sign in

This number can change until placing order has been complete.
Sign in to be able to leave a review.

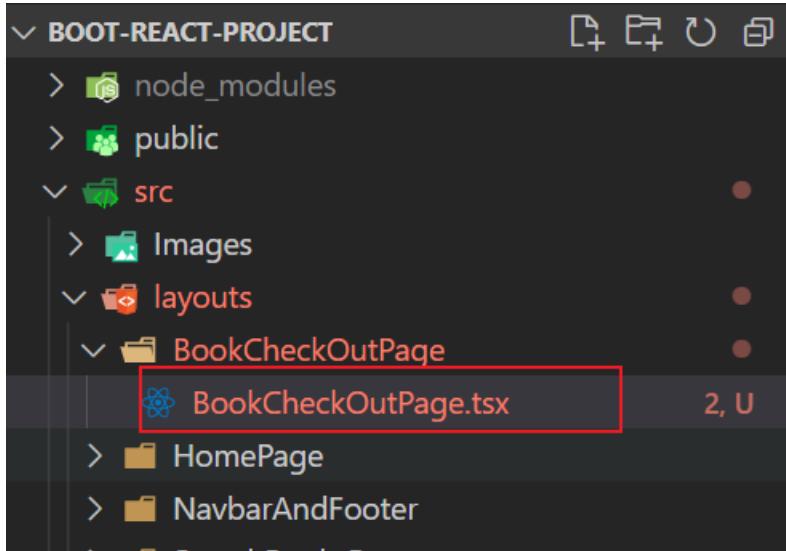
PART A.

BOOK CHECKOUT PAGE IMPLEMENTATION

Under “layouts” folder, create a new folder , name it “BookCheckOutPage”.



Inside this folder, create a new file “BookCheckOutPage.tsx”

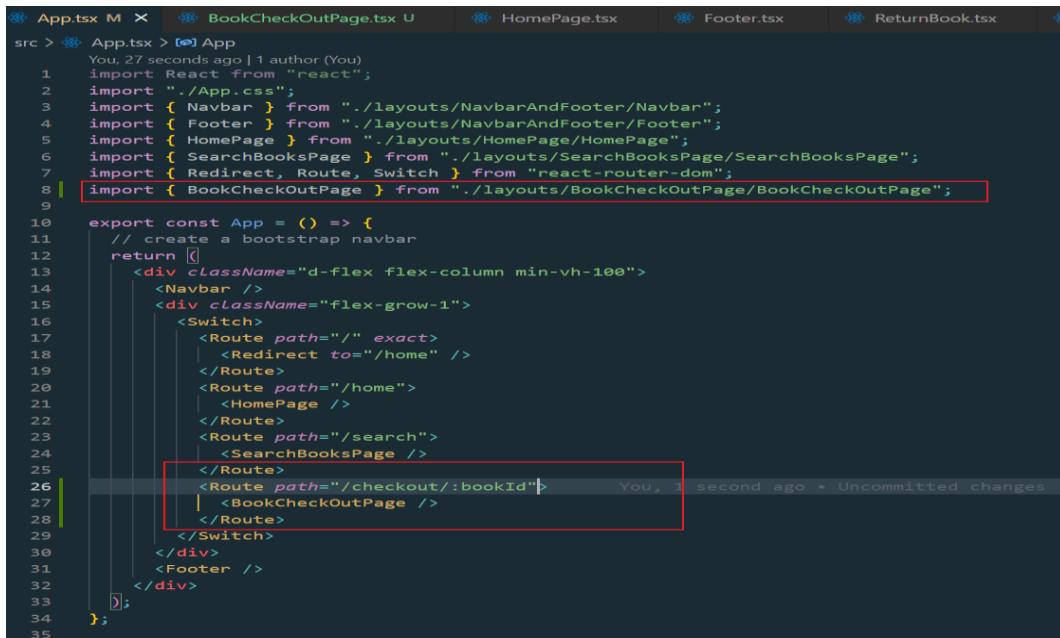


Write the code like this:



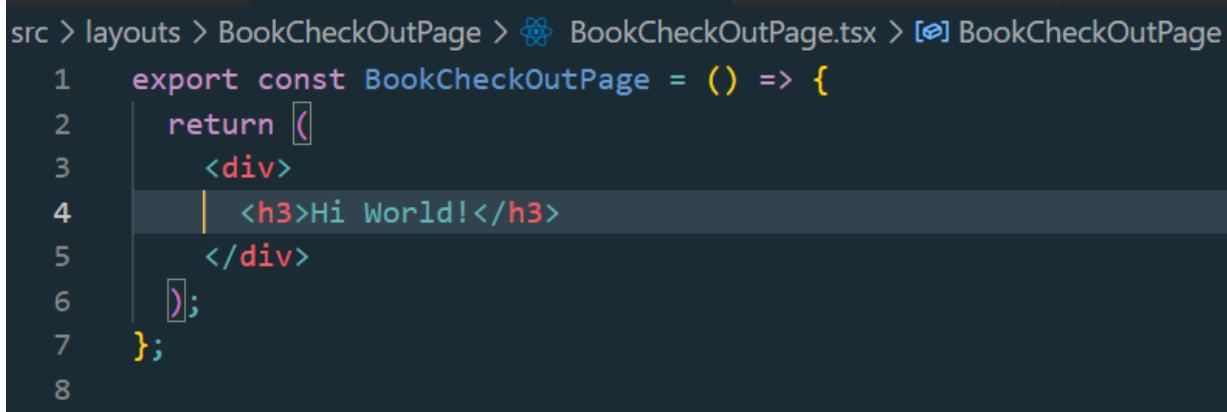
```
App.tsx BookCheckOutPage.tsx U X HomePage.tsx
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > ...
1  export const BookCheckOutPage = () => {
2    return <div></div>;
3  };
4
```

Go to App.tsx file, between Switch tag ,add a route to our BookCheckoutPage, and then the BookCheckoutPage component will automatically imported on the top of the page.



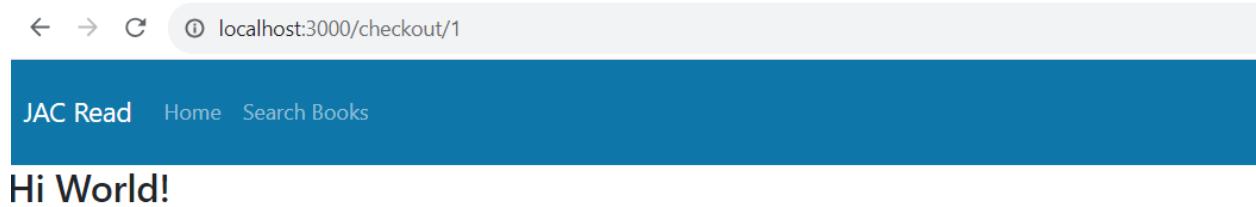
```
App.tsx M X BookCheckOutPage.tsx U HomePage.tsx Footer.tsx ReturnBook.tsx
src > App.tsx > App.tsx
1 import React from "react";
2 import "./App.css";
3 import { Navbar } from "./layouts/NavbarAndFooter/Navbar";
4 import { Footer } from "./layouts/NavbarAndFooter/Footer";
5 import { HomePage } from "./layouts/HomePage/HomePage";
6 import { SearchBooksPage } from "./layouts/SearchBooksPage/SearchBooksPage";
7 import { Redirect, Route, Switch } from "react-router-dom";
8 import { BookCheckOutPage } from "./layouts/BookCheckOutPage/BookCheckOutPage";
9
10 export const App = () => {
11   // create a bootstrap navbar
12   return [
13     <div className="d-flex flex-column min-vh-100">
14       <Navbar />
15       <div className="flex-grow-1">
16         <Switch>
17           <Route path="/" exact>
18             | <Redirect to="/home" />
19           </Route>
20           <Route path="/home">
21             | <HomePage />
22           </Route>
23           <Route path="/search">
24             | <SearchBooksPage />
25           </Route>
26           <Route path="/checkout/:bookId"> You, 27 seconds ago | 1 author (You)
27             | <BookCheckOutPage />
28           </Route>
29         </Switch>
30       </div>
31       <Footer />
32     </div>
33   ];
34 }
35
```

Go back to our BookCheckoutPage.tsx, add one line of code.



```
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
1  export const BookCheckOutPage = () => {
2    return (
3      <div>
4        | <h3>Hi World!</h3>
5        </div>
6    );
7  };
8
```

Run the application, open our browser, and visit <http://localhost:3000/checkout/1>, we will get the page like this: We have Nav bar, our h3 content and the footer, which means that our route works



In the BookCheckOutPage.tsx, we first create useState for book object, loading process , httpError and the book Id.

A screenshot of a code editor showing the 'BookCheckOutPage.tsx' file. The code uses React hooks to manage state for a book, its loading status, and an error. It also retrieves the book ID from the URL pathname. The code is annotated with a red box highlighting the state declarations and the book ID retrieval line.

Pathname: Localhost:3000/checkout/1 ← bookId
Index: 0 1 2

Create the useEffect method. Remember the useEffect method we created in the Carousel component? We can copy that part of code. The only difference is, here we are dealing with one single book not a book array.

```
src > layouts > HomePage > components > 📂 Carousel.tsx > 📁 Carousel
  export const Carousel = () => {
    // Create a book array
    const [books, setBooks] = useState<BookModel>([]);
    // For judging whether in loading
    const [isLoading, setIsLoading] = useState(true);
    // API call failure scenario
    const [httpError, setHttpError] = useState(null);
    useEffect(() => {
      const fetchBooks = async () => {
        const baseUrl: string = "http://localhost:8080/api/books";
        const url: string = `${baseUrl}?page=0&size=9`;
        // fetching the url data
        const response = await fetch(url);
        //failure scenario
        if (!response.ok) {
          throw new Error("Something went wrong!");
        }
        // transfer the url data into json
        const responseJson = await response.json();
        // get the data which is the object of embedded books
        const responseData = responseJson._embedded.books;
        // create a book array
        const loadedBooks: BookModel[] = [];
        // iterate the object in responseData, push them into the book array
        for (const key in responseData) {
          loadedBooks.push({
            id: responseData[key].id,
            title: responseData[key].title,
            author: responseData[key].author,
            description: responseData[key].description,
            copies: responseData[key].copies,
            copiesAvailable: responseData[key].copiesAvailable,
            category: responseData[key].category,
            img: responseData[key].img,
          });
        }
      };
      fetchBooks();
    }, []);
  }

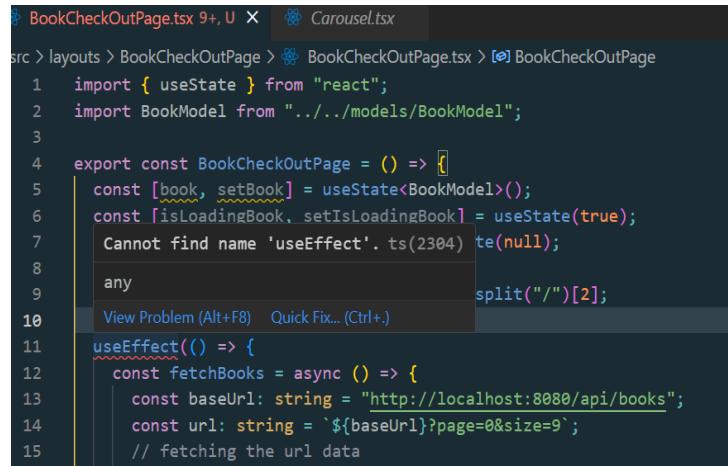
```

Back to BookCheckOutPage.tsx, paste the code we copied and put it just under the constant code part.

```
c > layouts > BookCheckOutPage > 📂 BookCheckOutPage.tsx > 📁 BookCheckOutPage
  1 ~ import { useState } from "react";
  2   import BookModel from "../../models/BookModel";
  3
  4 ~ export const BookCheckOutPage = () => {
  5   const [book, setBook] = useState<BookModel>();
  6   const [isLoadingBook, setIsLoadingBook] = useState(true);
  7   const [httpError, setHttpError] = useState(null);
  8
  9   const bookId = window.location.pathname.split("/")[2];
 10
 11 ~ useEffect(() => {
 12   const fetchBooks = async () => {
 13     const baseUrl: string = "http://localhost:8080/api/books";
 14     const url: string = `${baseUrl}?page=0&size=9`;
 15     // fetching the url data
 16     const response = await fetch(url);
 17     //failure scenario
 18     if (!response.ok) {
 19       throw new Error("Something went wrong!");
 20     }
 21     // transfer the url data into json
 22     const responseJson = await response.json();
 23     // get the data which is the object of embedded books
 24     const responseData = responseJson._embedded.books;
 25     // create a book array
 26     const loadedBooks: BookModel[] = [];
 27     // iterate the object in responseData, push them into the book array, and set
 28     for (const key in responseData) {
 29       loadedBooks.push({
 30         id: responseData[key].id,
 31         title: responseData[key].title,
 32         author: responseData[key].author,
 33         description: responseData[key].description,
 34         copies: responseData[key].copies,
 35         copiesAvailable: responseData[key].copiesAvailable,
 36       });
 37     }
 38   };
 39   fetchBooks();
 40 };

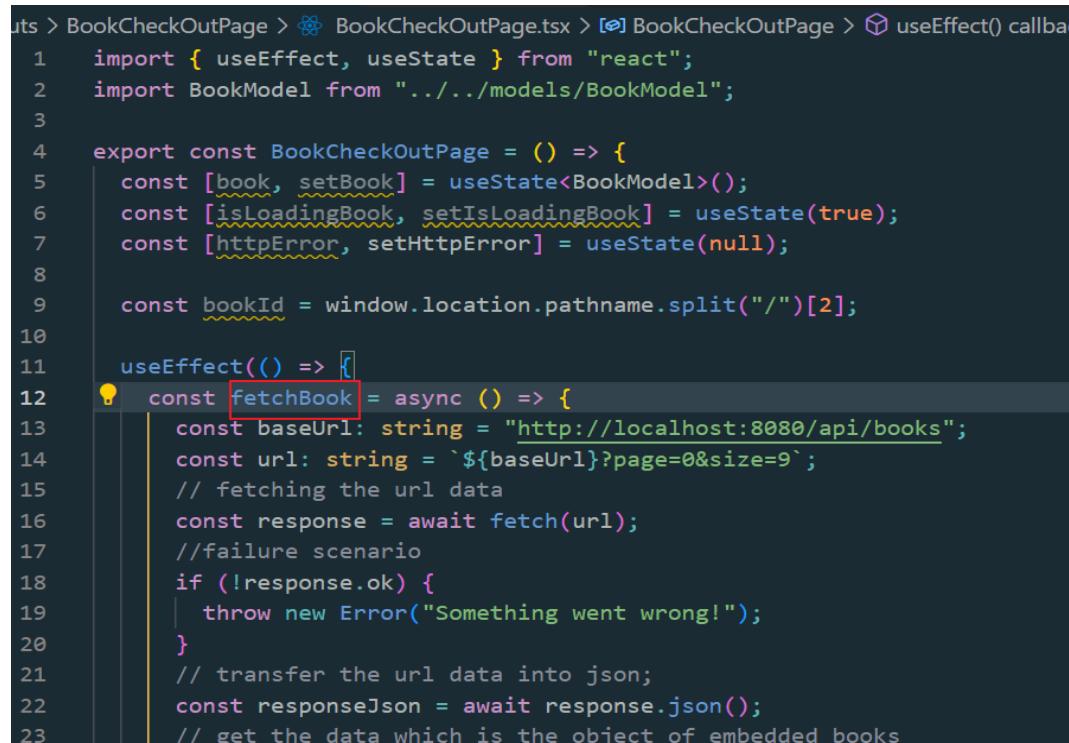
```

Begin to fix the red errors. Put the cursor on the red tilde, we will get the error description and a quick fix button.



```
BookCheckOutPage.tsx 9+ U ✘ Carouseltsx
src > layouts > BookCheckOutPage > ✘ BookCheckOutPage.tsx > BookCheckOutPage
1 import { useState } from "react";
2 import BookModel from "../../models/BookModel";
3
4 export const BookCheckOutPage = () => [
5   const [book, setBook] = useState<BookModel>();
6   const [isLoadingBook, setIsLoadingBook] = useState(true);
7   Cannot find name 'useEffect'. ts(2304) te(null);
8   any
9     split("/");
10    View Problem (Alt+F8) Quick Fix... (Ctrl+.)
11  useEffect(() => {
12    const fetchBooks = async () => {
13      const baseUrl: string = "http://localhost:8080/api/books";
14      const url: string = `${baseUrl}?page=0&size=9`;
15      // fetching the url data
```

refactor the constant fetchBooks to fetchBook. Because here we only deal with one single book. Right click the mouse, choose “rename symbol”, type new name ,it will change all the fetchBooks in the file.



```
ts > BookCheckOutPage > ✘ BookCheckOutPage.tsx > BookCheckOutPage > ✎ useEffect() callback
1 import { useEffect, useState } from "react";
2 import BookModel from "../../models/BookModel";
3
4 export const BookCheckOutPage = () => {
5   const [book, setBook] = useState<BookModel>();
6   const [isLoadingBook, setIsLoadingBook] = useState(true);
7   const [httpError, setHttpError] = useState(null);
8
9   const bookId = window.location.pathname.split("/")[2];
10
11  useEffect(() => {
12    const fetchBook = async () => {
13      const baseUrl: string = "http://localhost:8080/api/books";
14      const url: string = `${baseUrl}?page=0&size=9`;
15      // fetching the url data
16      const response = await fetch(url);
17      //failure scenario
18      if (!response.ok) {
19        throw new Error("Something went wrong!");
20      }
21      // transfer the url data into json;
22      const responseJson = await response.json();
23      // get the data which is the object of embedded books
```

fix url code. Use `` instead of "", and add bookId in the end, delete the url line.

```
useEffect(() => {
  const fetchBook = async () => [
    const baseUrl: string = `http://localhost:8080/api/books/${bookId}`;
    const url: string = `${baseUrl}?page=0&size=9`;
    // fetching the url data
    const response = await fetch(url);
    //failure scenario
    if (!response.ok) {
      throw new Error("Something went wrong!");
    }
}
```

In the await fetch (), pass the baseUrl we just fixed :

```
useEffect(() => {
  const fetchBook = async () => [
    const baseUrl: string = `http://localhost:8080/api/books/${bookId}`;
    // fetching the url data
    const response = await fetch(baseUrl);
    //failure scenario
    if (!response.ok) {
      throw new Error("Something went wrong!");
    }
}
```

delete the constant responseData which we used to get the whole books objects. Because we just use bookId to get one book object, we don't need to get the whole _embedded.books.

```
// transfer the url data into json;
const responseJson = await response.json();
// get the data which is the object of embedded books
const responseData = responseJson._embedded.books;
// create a book array
const loadedBooks: BookModel[] = [];
// iterate the object in responseData, push them into the book array, and set the loading process finished.
for (const key in responseData) {
  loadedBooks.push({
    id: responseData[key].id,
    title: responseData[key].title,
    author: responseData[key].author,
    description: responseData[key].description,
    copies: responseData[key].copies,
    copiesAvailable: responseData[key].copiesAvailable,
    category: responseData[key].category,
    img: responseData[key].img,
  });
}
```

For the same reason above, we need to change the constant “loadedBooks” (array) into “loadedBook” (single object) . set all the book’s properties into the constant . and delete the iteration code in the books array(responseData). Because here we just get one book object by its id. fix the “setBooks” error. Change the plural form books into Singular form, since here we only get one book object.

```
// create a book array
const loadedBook: BookModel = {
  id: responseJson.id,
  title: responseJson.title,
  author: responseJson.author,
  description: responseJson.description,
  copies: responseJson.copies,
  copiesAvailable: responseJson.copiesAvailable,
  category: responseJson.category,
  img: responseJson.img,
};

setBook(loadedBook);
setIsLoading(false);
};

fetchBook().catch((error: any) => {
  setIsLoading(false);
  setHttpError(error.message);
});
```

Go to Carousel.tsx file, copy the spinner Loading and http error part into our BookCheckoutPage.tsx.

```
52
53  if(isLoading) {      You, 2 days ago • Consumer
54    return <SpinnerLoading />;
55  }
56
57  if(httpError) {
58    return (
59      <div className="container m-5">
60        <p>{httpError}</p>
61      </div>
62    );
63  }
```

Just paste it after the fetchBook() catch error part.

```
37    };
38    fetchBook().catch((error: any) => {
39      setIsLoading(false);
40      setHttpError(error.message);
41    });
42  }, []);
43
44  if (isLoading) {
45    return <SpinnerLoading />;
46  }
47
48  if (httpError) [
49    return (
50      <div className="container m-5">
51        <p>{httpError}</p>
52      </div>
53    );
54  ]
55
56  return (
57    <div>
58      <h3>Hi World!</h3>
59    </div>
60  );
61};
```

fix the errors in the spinner Loading and http error part which we just pasted. Now, we finished all the useEffect part.

```
✓  if (isLoading) [
    return <SpinnerLoading />;
]

✓  if (httpError) {
✓    return (
✓      <div className="container m-5">
✓        <p>{httpError}</p>
✓      </div>
✓    );
}
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
1  import { useEffect, useState } from "react";
2  import BookModel from "../../models/BookModel";
3  import { SpinnerLoading } from "../Utils/SpinnerLoading";
4
```

Go to return part, we first create an div to display our book image on the left half of the page.

```
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
54  );
55
56
57  return (
58    <div>
59      <div className="container d-none d-lg-block">
60        <div className="row mt-5">
61          /* left side of the page: book image part */
62          <div className="col-sm-2 col-md-2">
63            {book?.img ? (
64              <img src={book?.img} width="226" height="349" alt="Book" />
65            ) : (
66              <img
67                src={require("../../Images/BooksImages/book-luv2code-1000.png")}
68                width="226"
69                height="349"
70                alt="Book"
71              />
72            )}
73        </div>
```

then we create another div to display our book info on the right half of the page.

```
74  /* right side of the page: book info part */
75  <div className="col-4 col-md-4 container">
76    <div className="ml-2">
77      <h2>{book?.title}</h2>
78      <h5 className="text-primary">{book?.author}</h5>
79      <p className="lead">{book?.description}</p>
80    </div>
81  </div>
82  <hr />
```

create book image and book info part for a mobile version page. Just change the wrapper div's className and copy the inside code from the web page version.

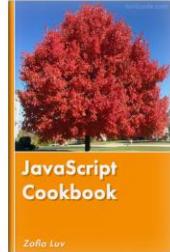
```
85  /* mobile version */
86  /* image part */
87  <div className="container d-lg-none mt-5">
88    <div className="d-flex justify-content-center align-items-center">
89      {book?.img ? (
90        <img src={book?.img} width="226" height="349" alt="Book" />
91      ) : (
92        <img
93          src={require("../../../../Images/BooksImages/book-luv2code-1000.png")}
94          width="226"
95          height="349"
96          alt="Book"
97        />
98      )}
99    </div>
100   /* book info part */
101  <div className="mt-4">
102    <div className="ml-2">
103      <h2>{book?.title}</h2>
104      <h5 className="text-primary">{book?.author}</h5>
105      <p className="lead">{book?.description}</p>
106    </div>
107  </div>
108  <hr />
109 </div>
110 </div>
```

Let's run the application. Input the url: <http://localhost:3000/checkout/1>. We can see the page.

localhost:3000/checkout/1

localhost:3000/checkout/1

Home Search Books



JavaScript Cookbook

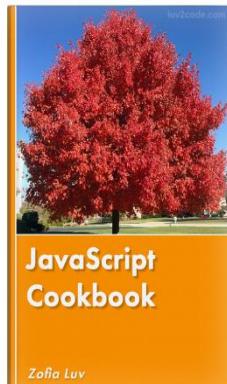
Luv, Zofia

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin risus tortor, condimentum eget sapien ac, dapibus varius ligula. Maecenas justo erat, semper sed nunc vel, vulputate eleifend dui. Integer id ipsum vitae nisi malesuada feugiat. Proin sit amet quam laoreet, feugiat mi vitae, vestibulum dui. Aliquam erat volutpat. Etiam hendrerit erat nec mi auctor elementum. Curabitur vestibulum lectus a ante tempor tincidunt et sed orci. Proin maximus tortor in risus auctor efficitur. Phasellus quam mauris, laoreet et feugiat ac, imperdiet at quam. Nullam sollicitudin nec diam vel finibus.

Mobile version:

← → C ① localhost:3000/ch... 📁 🔍 ⌂ ☆ ⚙ ⌂ Y :

AC Read



JavaScript Cookbook

Luv, Zofia

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin risus tortor, condimentum eget sapien ac, dapibus varius ligula. Maecenas justo erat, semper sed nunc vel, vulputate eleifend dui. Integer id ipsum vitae nisi malesuada feugiat. Proin sit amet quam laoreet, feugiat mi vitae, vestibulum dui. Aliquam erat volutpat. Etiam hendrerit erat nec mi auctor elementum. Curabitur vestibulum lectus a ante tempor tincidunt et sed orci. Proin maximus tortor in risus auctor

PART B. BOOK CHECKOUT LINK

We will fix all the links to our detailed book page(Checkout page).

Step 1, Go to "SearchBook.tsx", scroll down to the bottom, find the "view details" link.

The screenshot shows a code editor with a file tree on the left and the content of the selected file on the right.

File Tree:

- Explorer
- OT-REACT-PROJ... (1)
- node_modules
- public
- src
 - Images
 - layouts
 - BookCheckOutPage
 - BookCheckOutP... 1, U
 - HomePage
 - components
 - Carousel.tsx
 - ExploreTopBooks.tsx
 - Heros.tsx
 - LibraryServices.tsx
 - ReturnBook.tsx
 - HomePage.tsx
 - NavbarAndFooter
 - SearchBooksPage
 - components
 - SearchBook.tsx 1
 - Utils
 - models
 - App.css

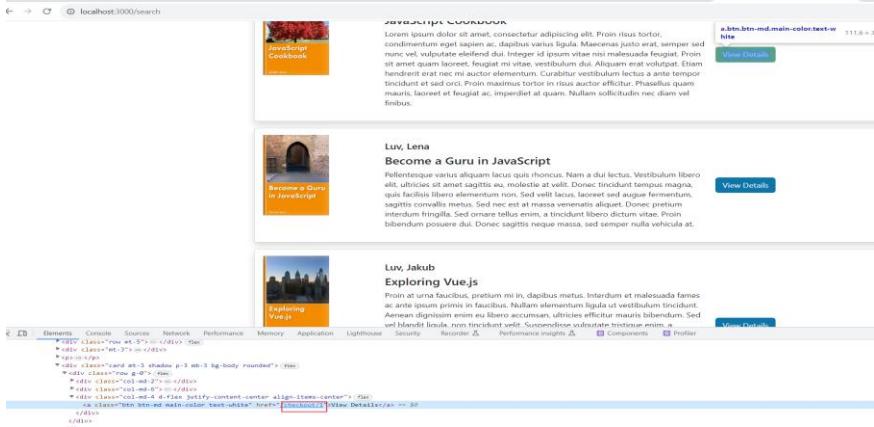
Code View (SearchBook.tsx):

```
22      <img src={props.book.img} width="123" height="196" alt="Book" />
23    ) : (
24      25        <img
25          src={require("../Images/BooksImages/book-luv2code-1000.png")}
26          width="123"
27          height="196"
28          alt="Book"
29          />
30    )
31  </div>
32  </div>
33  <div className="col-md-6">
34    <div className="card-body">
35      <h5 className="card-title">{props.book.author}</h5>
36      <h4>{props.book.title}</h4>
37      <p className="card-text">{props.book.description}</p>
38    </div>
39  </div>
40  <div className="col-md-4 d-flex justify-content-center align-items-center">
41    <a className="btn btn-md main-color text-white" href="#">
42      View Details
43    </a>
44  </div>
45  </div>
46  </div>
47  );
48 };
```

Change "a" to "Link", "href" to "to", vaule of href "#" to the value of {'/checkout/\${props.book.id}'}

```
/* btn to the details page */
<div className="col-md-4 d-flex justify-content-center align-items-center">
  <Link
    className="btn btn-md main-color text-white"
    to={`/checkout/${props.book.id}`}
  >
    View Details
  </Link>
</div>
</div>
};
```

Now, run the application, open our search page. For the first book, check its “view details” button in the console. You will find that its href equals to “/checkout/1”. It means now the app gets its Id.



If we click on the “View Details” button, it redirects to value of the href we have just checked, shows the following page,



Crash Course in Big Data

Luv, Judy

Morbi eu tempus eros, in imperdiet sem. Nulla sed sagittis nisl, porttitor fringilla libero. Nullam ut urna aliquet, hendrerit quam in, dignissim diam. In in nibh vel nisi fermentum pretium sit amet vitae mi. Pellentesque eget augue efficitur, volutpat tellus eget, fringilla augue. Pellentesque tempus mi ac risus lacinia, et tincidunt lectus rutrum. Nullam et nibh a odio luctus tincidunt nec in ipsum. Sed a est nulla. Nulla purus turpis, dignissim sit amet euismod lobortis, consequat ut dul. Maecenas commodo velit in elementum placerat. Nam sit amet blandit ante, sit amet mollis neque. Ut placerat venenatis leo sit amet dapibus. Nunc varius cursus lobortis. Aenean euismod dul at diam euismod aliquet. Fusce feugiat orci nec commodo placerat.

PART C. STAR REVIEW COMPONENT

Google “bootstrap 5 icon.” Now, we will use bootstrap icon to display our star reviews. Since we use bootstrap version 5, google bootstrap 5 icon.

Google

bootstrap 5 icon

Images CDN Size Color W3schools Input Button Search box Link

About 132,000,000 results (0.29 seconds)

 Get Bootstrap Icons
<https://icons.getbootstrap.com> :

Bootstrap Icons · Official open source SVG icon library for ...

Official open source **icon** library for **Bootstrap**. ... Free, high quality, open source **icon** library with over 1,800 **icons**. ... **5-circle-fill** **5-square**.

You've visited this page 2 times. Last visit: 2023-08-10

Icons
Bootstrap Icons is a growing library of SVG icons that are designed ...

Search
Official open source SVG icon library for Bootstrap. ... 16 16 ...

Download
Official open source SVG icon library for Bootstrap. ... class="bi ..."

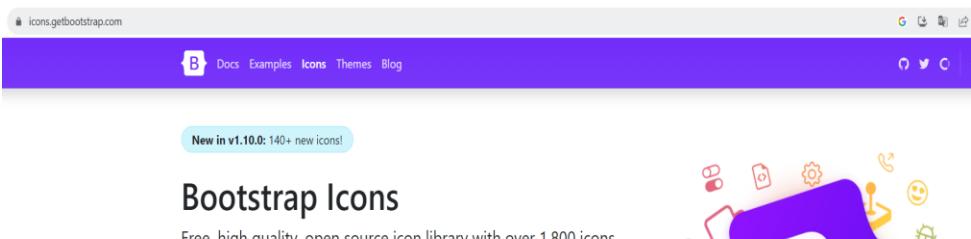
List
Official open source SVG icon library for Bootstrap. ... bi-list ...

[More results from getbootstrap.com »](#)

 Get Bootstrap

click the link : <http://icons.getbootstrap.com>

icons.getbootstrap.com



The screenshot shows the Bootstrap Icons website. At the top, there's a purple header bar with the Bootstrap logo and navigation links for Docs, Examples, Icons, Themes, and Blog. Below the header, a teal box highlights "New in v1.10.0: 140+ new icons!". The main title "Bootstrap Icons" is in large, bold, black font. Below it, a paragraph describes the library as a free, high-quality, open-source icon library with over 1,800 icons, available as SVGs, SVG sprite, or web fonts, and can be used with or without Bootstrap. There are two buttons at the bottom left: one for running "npm i bootstrap-icons" and another for "Open in Figma". To the right, there's a large purple icon of a white letter "B" surrounded by various colorful, rounded icons representing different functions like file, gear, and media.

New in v1.10.0: 140+ new icons!

Bootstrap Icons

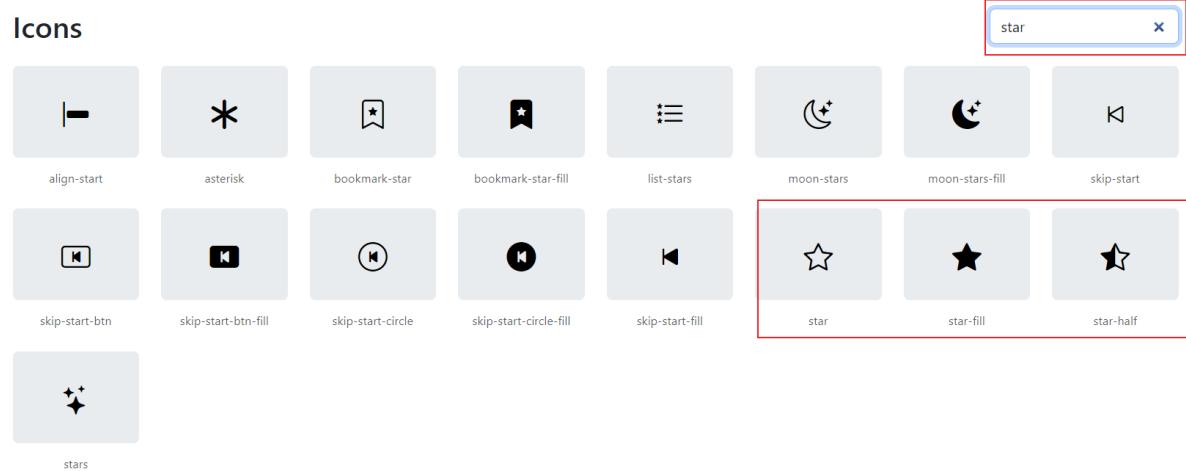
Free, high quality, open source icon library with over 1,800 icons. Include them anyway you like—SVGs, SVG sprite, or web fonts. Use them with or without [Bootstrap](#) in any project.

\$ npm i bootstrap-icons

[Open in Figma](#)

 Help the Carbon Ads Network display ads right here for services & products you actually want.
ads via Carbon

In the search box, input “star”. It will pop out many options. In our App, we use empty star, star-filled and half star.



Click on the star-filled go to the main page of “star-fill”. Scroll down the page, click on the copy sign in the “Copy HTML” part.

A large black five-pointed star icon is centered on a white background with a thin gray border. To the right of the star, there is a "Download" section with a "Download SVG" button. Below that is an "Icon font" section with a code snippet and a copy icon.

Examples

The screenshot displays a collection of UI elements featuring stars:

- A large black star icon followed by the text "Heading".
- A smaller black star icon followed by the text "Smaller heading".
- The text "Inline text" followed by a black star icon.
- The text "Example link text" followed by a black star icon.
- Three buttons: a blue one labeled "Button", a green one labeled "Button", and a red one labeled "Button".
- A row of five gray star icons of decreasing size from left to right.
- A form input group containing a gray star icon and the placeholder text "Input group example".

Download

Download the SVG to use or edit.

[Download SVG](#)

Icon font

Using the web font? Copy, paste, and go.

```
<i class="bi bi-star-fill"></i>
```

Code point

Unicode: U+F586

CSS: \F586

JS: \uF586

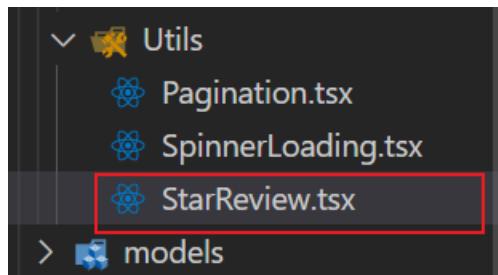
HTML: 

Copy HTML

Paste the SVG right into your project's code.

```
<svg xmlns="http://www.w3.org/2000/svg" wid |  
  <path d="M3.612 15.443c-.386.198-.824-.149-. /  
</svg>
```

Back to our VS Code. Create a new file under folder “Utils”, name it “StarsReview.tsx”.



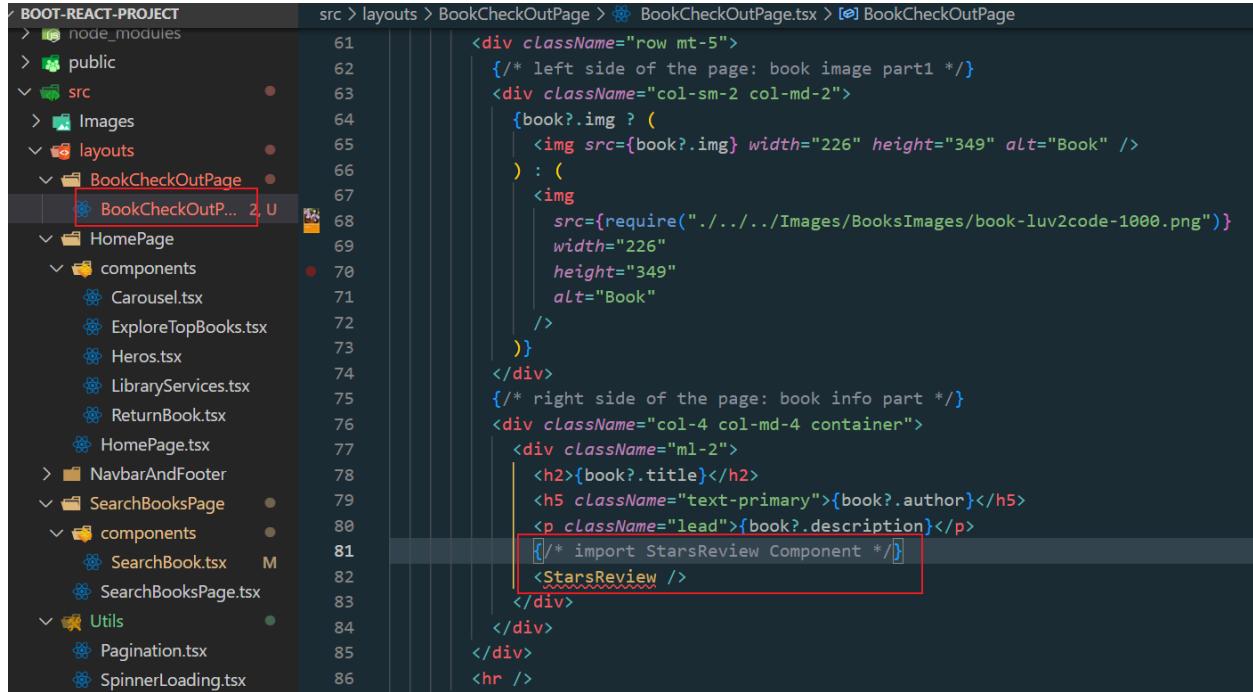
In the StarsReview.tsx, Write the code, paste the code we copied from the bootstrap icon website in the parentheses after “return”. We will get red error. Put the cursor on the red tilde, we can see the error description.

```
src > layouts > Utils > StarsReview.tsx > StarsReview
1  export const StarsReview: React.FC<{ Rating: number; size: number }> = (
2    props
3  ) => {
4    return (
5      <div>
6        <s>` is not assignable to type 'SVGProps<SVGSVGElement>'.
7          Property 'class' does not exist on type 'SVGProps<SVGSVGElement>'. Did you mean 'className'? ts(2322)
8
9        (property) class: string
10       View Problem (Alt+F8) Quick Fix... (Ctrl+.)
11       class="bi bi-star-fill"
12       viewBox="0 0 16 16"
13     >
14       <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.765c-.329-.314-.158-.888.283-.951L4.898-.696L7.538.792c.197-.39.73-
15     </svg>
16   </div>
17 );
18 };
19 
```

Fix the issue by click on the “Quick Fix”.

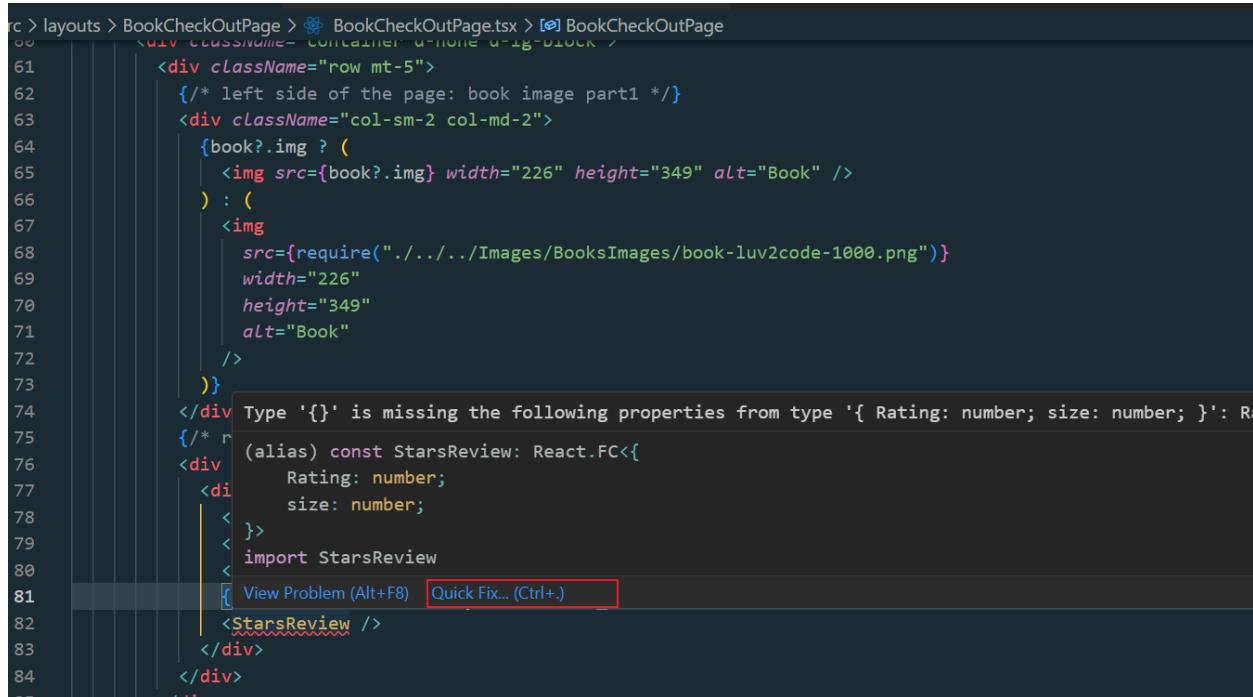
```
: > layouts > Utils > StarsReview.tsx > StarsReview
1  export const StarsReview: React.FC<{ Rating: number; size: number }> = (
2    props
3  ) => {
4    return (
5      <div>
6        <svg
7          xmlns="http://www.w3.org/2000/svg"
8          width="16"
9          height="16"
10         fill="currentColor"
11         className="bi bi-star-fill"
12         viewBox="0 0 16 16"
13       >
14         <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.765c-.329-.314-.158-
15       </svg>
16     </div>
17   );
18 };
19 
```

Go to BookCheckOutPage.tsx, import the StarsReview component we created just after the book description part. Fix the issue by import StartsReview Component.



```
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
61     <div className="row mt-5">
62         /* left side of the page: book image part1 */
63         <div className="col-sm-2 col-md-2">
64             {book?.img ? (
65                 <img src={book?.img} width="226" height="349" alt="Book" />
66             ) : (
67                 <img
68                     src={require("../../../../Images/BooksImages/book-luv2code-1000.png")}
69                     width="226"
70                     height="349"
71                     alt="Book"
72                 />
73             )}
74         </div>
75         /* right side of the page: book info part */
76         <div className="col-4 col-md-4 container">
77             <div className="ml-2">
78                 <h2>{book?.title}</h2>
79                 <h5 className="text-primary">{book?.author}</h5>
80                 <p className="lead">{book?.description}</p>
81             /* import StarsReview Component */
82             <StarsReview />
83         </div>
84     </div>
85     </div>
86     <hr />
```

Fix the error. Our StarsReview component show an error. Right click the mouse, get the error description. We can see it's because our component is missing properties it's supposed to have. Click on the "Quick Fix" or add the rating and size properties directly.



```
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
61     <div className="row mt-5">
62         /* left side of the page: book image part1 */
63         <div className="col-sm-2 col-md-2">
64             {book?.img ? (
65                 <img src={book?.img} width="226" height="349" alt="Book" />
66             ) : (
67                 <img
68                     src={require("../../../../Images/BooksImages/book-luv2code-1000.png")}
69                     width="226"
70                     height="349"
71                     alt="Book"
72                 />
73             )}
74         </div>
75         /* right side of the page: book info part */
76         <div className="col-4 col-md-4 container">
77             <div className="ml-2">
78                 <h2>{book?.title}</h2>
79                 <h5 className="text-primary">{book?.author}</h5>
80                 <p className="lead">{book?.description}</p>
81             <div>Type '{}' is missing the following properties from type '{ Rating: number; size: number; }': Rating, size</div>
82             <div>(alias) const StarsReview: React.FC<{
83                 Rating: number;
84                 size: number;
85             }></div>
86             <div><StarsReview /></div>
87             <div><StarsReview /></div>
88         </div>
89     </div>
90     <hr />
```

We try to add “rating={4}, size ={32}”

```
75      /* right side of the page: book info part */  
76      <div className="col-4 col-md-4 container">  
77          <div className="ml-2">  
78              <h2>{book?.title}</h2>  
79              <h5 className="text-primary">{book?.author}</h5>  
80              <p className="lead">{book?.description}</p>  
81              /* import StarsReview Component */  
82              <StarsReview rating={4} size={32} />  
83          </div>  
84      </div>
```

Run the application, go to the browser, input <http://localhost:3000/checkout/5> Go to the book whose id is 5. We get one full star review.

The screenshot shows a web browser window with the URL [@ localhost:3000/checkout/5](http://localhost:3000/checkout/5) in the address bar. The page content is as follows:

- Header: Home Search Books
- Book Cover: An orange book cover titled "Crash Course in Big Data" by "Judy Luv". The cover features a photograph of a path through a wooded area.
- Book Title: Crash Course in Big Data
- Author: Luv, Judy
- Description: A long paragraph of placeholder text (Lorem ipsum) describing the book.
- Rating: A red-bordered `<StarsReview rating={4} size={32} />` component, which displays a single filled star.

Go to StarsReview.tsx, change the value of width and height in the “svg” tag. And after className part, add style = {{color:“gold”}}. We get a one-star review under the book description. But there are still two problems to solve. One is parameter we passed does not work. We want rating ={4} and size={ 32} and it does not work. The other is we want to have a gold star in our App. Here we solve the size and the color problem. Values are changed as below:

```
src > layouts > Utils > ⚙️ StarsReview.tsx > 🏷️ StarsReview
1  export const StarsReview: React.FC<{ rating: number; size: number }> = (
2    props
3  ) => {
4    return (
5      <div>
6        <svg
7          xmlns="http://www.w3.org/2000/svg"
8          width={props.size}
9          height={props.size}
10         fill="currentColor"
11         className="bi bi-star-fill"
12         style={{ color: "gold" }}
13         viewBox="0 0 16 16"
14       >
15         <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.765c-.329-.511-.329-.511-.329-.511z"/>
16       </svg>
17     </div>
18   );
19 };
20 
```

Let's go back to our browser. Now we have a golden star review.

localhost:3000/checkout/5



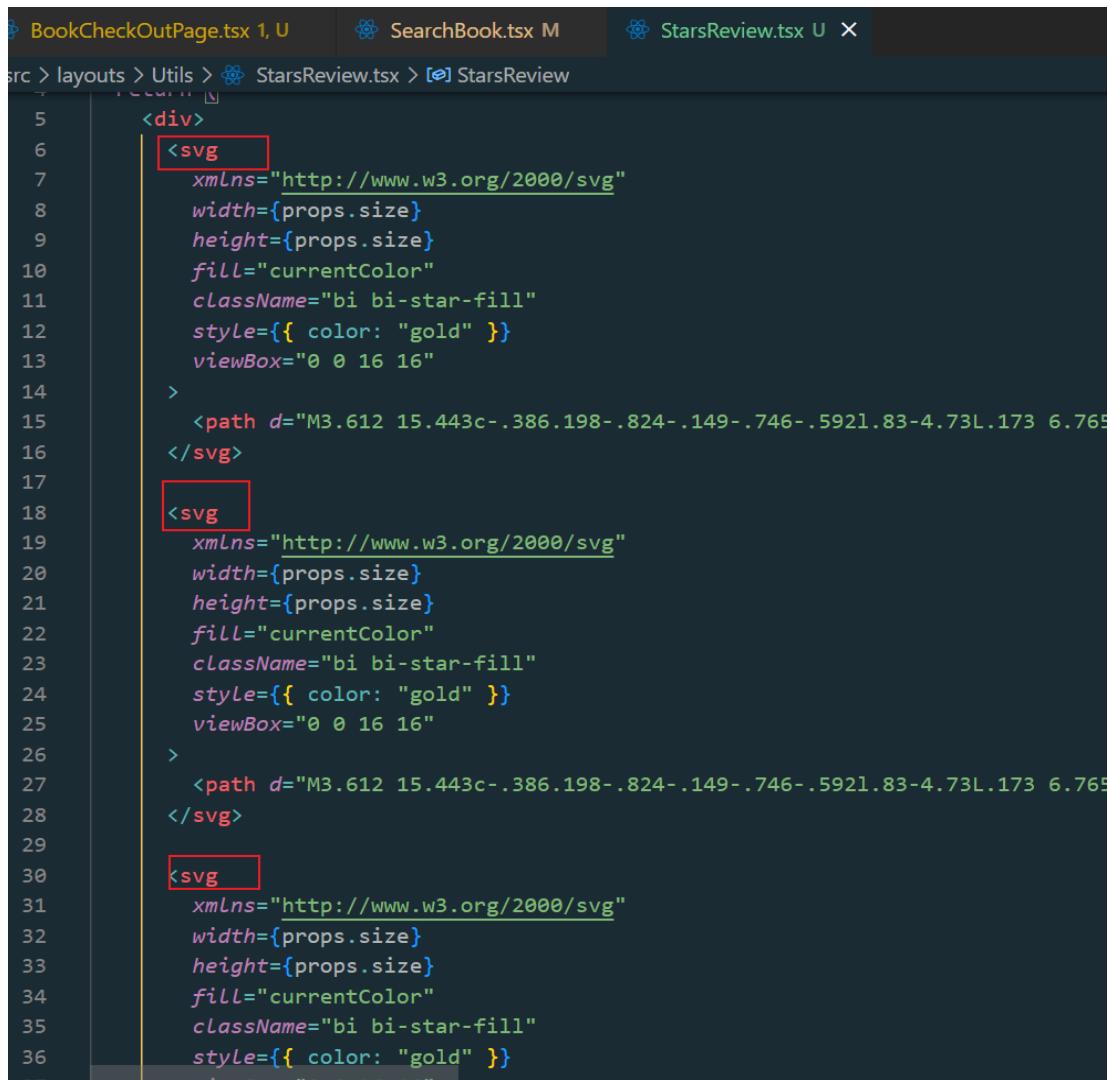
Crash Course in Big Data

Luv, Judy

Morbi eu tempus eros, in imperdiet sem. Nulla sed sagittis nisl, porttitor fringilla libero. Nullam ut urna aliquet, hendrerit quam in, dignissim diam. In in nibh vel nisi fermentum pretium sit amet vitae mi. Pellentesque eget augue efficitur, volutpat tellus eget, fringilla augue. Pellentesque tempus mi ac risus lacinia, et tincidunt lectus rutrum. Nullam et nibh a odio luctus tincidunt nec in ipsum. Sed ac est nulla. Nulla purus turpis, dignissim sit amet euismod lobortis, consequat ut dui. Maecenas commodo velit in elementum placerat. Nam sit amet blandit ante, sit amet mollis neque. Ut placerat venenatis leo sit amet dapibus. Nunc varius cursus lobortis. Aenean euismod dui at diam euismod aliquet. Fusce feugiat orci nec commodo placerat.



Now, if we copy “star fill” svg tag three times ,



```
BookCheckOutPage.tsx 1, U SearchBook.tsx M StarsReview.tsx U X
src > layouts > Utils > StarsReview.tsx > [o] StarsReview
5   <div>
6     <svg
7       xmlns="http://www.w3.org/2000/svg"
8       width={props.size}
9       height={props.size}
10      fill="currentColor"
11      className="bi bi-star-fill"
12      style={{ color: "gold" }}
13      viewBox="0 0 16 16"
14    >
15      <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.765
16    </svg>
17
18    <svg
19      xmlns="http://www.w3.org/2000/svg"
20      width={props.size}
21      height={props.size}
22      fill="currentColor"
23      className="bi bi-star-fill"
24      style={{ color: "gold" }}
25      viewBox="0 0 16 16"
26    >
27      <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.765
28    </svg>
29
30    <svg
31      xmlns="http://www.w3.org/2000/svg"
32      width={props.size}
33      height={props.size}
34      fill="currentColor"
35      className="bi bi-star-fill"
36      style={{ color: "gold" }}
```

Then copy “half star ” svg tag one time,

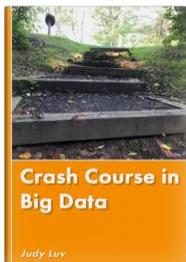


```
42 <svg
43   xmlns="http://www.w3.org/2000/svg"
44   width={props.size}
45   height={props.size}
46   fill="currentColor"
47   className="bi bi-star-half"
48   style={{ color: "gold" }}
49   viewBox="0 0 16 16"
50 >
51   <path d="M5.354 5.119 7.538.792A.516.516 0 0 1 8 .5c.183 0 .366.097.465.292l2.184
52 </svg>
```

Then copy “star ” svg tag one time,

```
<svg
  xmlns="http://www.w3.org/2000/svg"
  width={props.size}
  height={props.size}
  fill="currentColor"
  className="bi bi-star"
  style={{ color: "gold" }}
  viewBox="0 0 16 16"
>
  <path d="M2.866 14.85c-.078.444.36.791.746.593l4.39-2.256 4.389 2.256c.386.198.82
</svg>
```

Refresh the page <http://localhost:3000/checkout/2>, We will see a 3.5 golden star review, which equal to the value of each SVG tag’s occurrences.



Crash Course in Big Data

Luv, Judy

Morbi eu tempus eros, in imperdiet sem. Nulla sed sagittis nisl, porttitor fringilla libero. Nullam ut urna aliquet, hendrerit quam in, dignissim diam. In in nibh vel nisi fermentum pretium sit amet vitae mi. Pellentesque eget augue efficitur, volutpat tellus eget, fringilla augue. Pellentesque tempus mi ac risus lacinia, et tincidunt lectus rutrum. Nullam et nibh a odio luctus tincidunt nec in ipsum. Sed ac est nulla. Nulla purus turpis, dignissim sit amet euismod lobortis, consequat ut dui. Maecenas commodo velit in elementum placerat. Nam sit amet blandit ante, sit amet mollis neque. Ut placerat venenatis leo sit amet dapibus. Nunc varius cursus lobortis. Aenean euismod dui at diam euismod aliquet. Fusce feugiat orci nec commodo placerat.



Now, we create an array for each kind of star, and control the number of each kind of star by the length of the array. Let the length of the star fill be 3. Copy the original svg tag into our array function.

```
4  return (
5    <div>
6      {Array.from({ length: 3 }, (_, i) => (
7        <svg
8          key={i}
9          xmlns="http://www.w3.org/2000/svg"
10         width={props.size}
11         height={props.size}
12         fill="currentColor"
13         className="bi bi-star-fill"
14         style={{ color: "gold" }}
15         viewBox="0 0 16 16"
16       >
17         <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.765c-.329-.314-.158-.888.283-
18       </svg>
19     ))}
```

The same process for the half star and empty star, make the length of these two arrays be 1. So now, we should also have a 3.5 stars review.

```
22  {Array.from({ length: 1 }), (_, i) => (
23    // half star
24    <svg
25      xmlns="http://www.w3.org/2000/svg"
26      width={props.size}
27      height={props.size}
28      fill="currentColor"
29      className="bi bi-star-half"
30      style={{ color: "gold" }}
31      viewBox="0 0 16 16"
32    >
33      <path d="M5.354 5.119 7.538.792A.516.516 0 0 1 8 .5c.183 0 .366.097.465.292l2.184 4.327 4.898.69"/>
34    </svg>
35  )})
36
37  {Array.from({ length: 1 }, (_, i) => (
38    // star
39    <svg
40      xmlns="http://www.w3.org/2000/svg"
41      width={props.size}
42      height={props.size}
43      fill="currentColor"
44      className="bi bi-star"
45      style={{ color: "gold" }}
46      viewBox="0 0 16 16"
47    >
48      <path d="M2.866 14.85c-.078.444.36.791.746.593l4.39-2.256 4.389 2.256c.386.198.824-.149.746-.59"/>
49    </svg>
50  )})
51  </div>
52 );
```

Open our browser, we still see a 3.5 golden star review, nothing changes.



Crash Course in Big Data

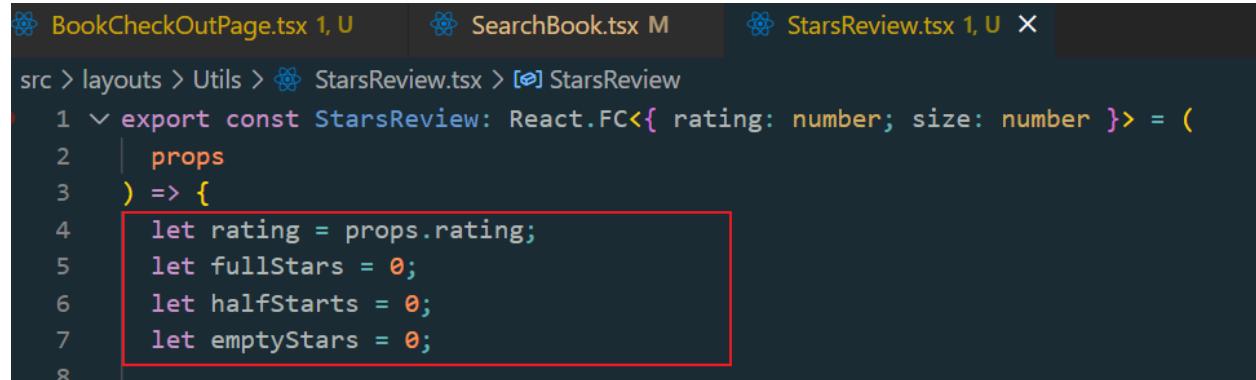
Luv, Judy

Morbi eu tempus eros, in imperdiet sem. Nulla sed sagittis nisl, porttitor fringilla libero. Nullam ut urna aliquet, hendrerit quam in, dignissim diam. In in nibh vel nisi fermentum pretium sit amet vitae mi. Pellentesque eget augue efficitur, volutpat tellus eget, fringilla augue. Pellentesque tempus mi ac risus lacinia, et tincidunt lectus rutrum. Nullam et nibh a odio luctus tincidunt nec in ipsum. Sed ac est nulla. Nulla purus turpis, dignissim sit amet euismod lobortis, consequat ut dui. Maecenas commodo velit in elementum placerat. Nam sit amet blandit ante, sit amet mollis neque. Ut placerat venenatis leo sit amet dapibus. Nunc varius cursus lobortis. Aenean euismod dui at diam euismod aliquet. Fusce feugiat orci nec commodo placerat.



Now, our stars reviews has nothing related with our rating parameter. We will implement the code to make our star reviews change dynamically with the rating value.

In the Utils folder, StarsReviews.tsx, we initialize four parameter instances: rating, fullStars, halfStars, emptyStars.



```
src > layouts > Utils > StarsReview.tsx [StarsReview]
1 export const StarsReview: React.FC<{ rating: number; size: number }> = (
2   | props
3   ) => {
4     let rating = props.rating;
5     let fullStars = 0;
6     let halfStarts = 0;
7     let emptyStars = 0;
8 }
```

Implement the logic relationship between rating and number of each kind of star. Calculate how many stars we have for each kind of star.

```
9 if (rating !== undefined && rating > 0 && rating <= 5) {
10   for (let i = 0; i <= 4; i++) {
11     if (rating - 1 >= 0) {
12       // calculate how many full stars we can have
13       fullStars = fullStars + 1;
14       rating = rating - 1;
15     } else if (rating === 0.5) {
16       // calculate how many half stars we can have
17       halfStarts = halfStarts + 1;
18       rating = rating - 0.5;
19     } else if (rating === 0) {
20       // calculate how many empty stars we can have
21       emptyStars = emptyStars + 1;
22     } else {
23       break;
24     }
25   }
26 } else {
27   emptyStars = 5;
28 }
```

change the static value of array length of each kind of star to a dynamical value

```
51
52   {fullStars}
53   {Array.from({ length: fullStars }, (_, i) => (
54     // star fill
55     <svg
56       key={i}
57       xmlns="http://www.w3.org/2000/svg"
58       width={props.size}
59       height={props.size}
60       fill="currentColor"
61       className="bi bi-star-fill"
62       style={{ color: "gold" }}
63       viewBox="0 0 16 16"
64     >
65       <path d="M3.612 15.443c-.386.198-.824-.149-.746-.592l.83-4.73L.173 6.7
66     </svg>
67   ))}
```



```
68   {halfStarts}
69   {Array.from({ length: halfStarts }, (_, i) => (
70     // half star
71     <svg
72       key={i}
73       xmlns="http://www.w3.org/2000/svg"
74       width={props.size}
75       height={props.size}
76       fill="currentColor"
77       className="bi bi-star-half"
78       style={{ color: "gold" }}
79       viewBox="0 0 16 16"
80     >
81       <path d="M5.354 5.119 7.538.792A.516.516 0 0 1 8 .5c.183 0 .366.097.46
82     </svg>
83   ))}
```



```
84   {emptyStars}
85   {Array.from({ length: emptyStars }, (_, i) => (
86     // star
87   ))}
```

Let's check if our code works. Go back to BookCheckoutPage, in the StarsReviews component, change the value of the rating to 2.5.

```
src > layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
  <div className="row mt-5">
    /* left side of the page: book image part1 */
    <div className="col-sm-2 col-md-2">
      {book?.img ? (
        <img src={book?.img} width="226" height="349" alt="Book" />
      ) : (
        <img
          src={require("../../../../Images/BooksImages/book-luv2code-1000.png")}
          width="226"
          height="349"
          alt="Book"
        />
      )}
    </div>
    /* right side of the page: book info part */
    <div className="col-4 col-md-4 container">
      <div className="ml-2">
        <h2>{book?.title}</h2>
        <h5 className="text-primary">{book?.author}</h5>
        <p className="lead">{book?.description}</p>
        {/* import StarsReview Component */}
        <StarsReview rating=[2.5] size={32} />
      </div>
    </div>
  </div>
  <hr />
```

Run the application, we get a 1.5 golden star here. Which means our code works perfectly.



Crash Course in Big Data

Luv, Judy

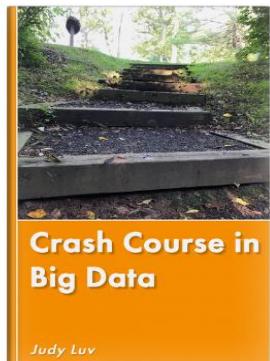
Morbi eu tempus eros, in imperdiet sem. Nulla sed sagittis nisl, porttitor fringilla libero. Nullam ut urna aliquet, hendrerit quam in, dignissim diam. In in nibh vel nisi fermentum pretium sit amet vitae mi. Pellentesque eget augue efficitur, volutpat tellus eget, fringilla augue. Pellentesque tempus mi ac risus lacinia, et tincidunt lectus rutrum. Nullam et nibh a odio luctus tincidunt nec in ipsum. Sed ac est nulla. Nulla purus turpis, dignissim sit amet euismod lobortis, consequat ut dui. Maecenas commodo velit in elementum placerat. Nam sit amet blandit ante, sit amet mollis neque. Ut placerat venenatis leo sit amet dapibus. Nunc varius cursus lobortis. Aenean euismod dui at diam euismod aliquet. Fusce feugiat orci nec commodo placerat.



import our StarsReview component in our mobile version code

```
3     /* mobile version */
4     /* image part */
5     <div className="container d-lg-none mt-5">
6       <div className="d-flex justify-content-center align-items-center">
7         {book?.img ? (
8           <img src={book?.img} width="226" height="349" alt="Book" />
9         ) : (
10           <img
11             src={require("../../../../Images/BooksImages/book-luv2code-1000.png")}
12             width="226"
13             height="349"
14             alt="Book"
15           />
16         )
17       )
18     </div>
19     /* book info part */
20     <div className="mt-4">
21       <div className="ml-2">
22         <h2>{book?.title}</h2>
23         <h5 className="text-primary">{book?.author}</h5>
24         <p className="lead">{book?.description}</p>
25         /* import StarsReview Component */
26         <StarsReview rating={2.5} size={32} />
27       </div>
28     </div>
```

Go to our mobile version application, Under the description of the book, there is also a 2.5 golden star review.



Crash Course in Big Data

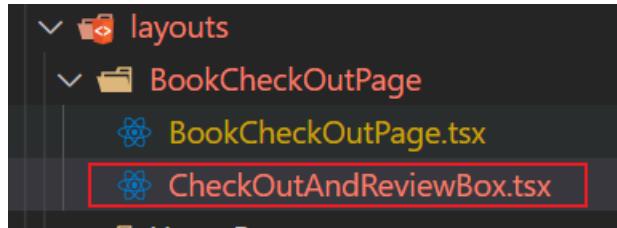
Luv, Judy

Morbi eu tempus eros, in imperdiet sem. Nulla sed sagittis nisl, porttitor fringilla libero. Nullam ut urna aliquet, hendrerit quam in, dignissim diam. In in nibh vel nisi fermentum pretium sit amet vitae mi. Pellentesque eget augue efficitur, volutpat tellus eget, fringilla augue. Pellentesque tempus mi ac risus lacinia, et tincidunt lectus rutrum. Nullam et nibh a odio luctus tincidunt nec in ipsum. Sed ac est nulla. Nulla purus turpis, dignissim sit amet euismod lobortis, consequat ut dui. Maecenas commodo velit in elementum placerat. Nam sit amet blandit ante, sit amet mollis neque. Ut placerat venenatis leo sit amet dapibus. Nunc varius cursus lobortis. Aenean euismod dui at diam euismod aliquet. Fusce feugiat orci nec commodo placerat.



PART D. CHECKOUT COMPONENT

Go to folder BookCheckoutPage, right click and choose “New file” to create a new file.name it “CheckoutAndReviewBox.tsx”.



as before, we create export const structure. Pass the “book” which is BookModel object as props.

```
src > layouts > BookCheckOutPage > CheckoutAndReviewBox.tsx > CheckoutAndReviewBox
1 import BookModel from "../../models/BookModel";
2
3 export const CheckoutAndReviewBox: React.FC<{book: BookModel | undefined, mobile: boolean}> = (props) =>{
4   return (
5     ...
6   );
7 }
8
```

In “return” statement, create the layout for the page. As it’s a responsive page, we add another props “mobile” in the constant CheckoutAnsReviewBox.

```
return (
  // if in mobile version, use "card d-flex mt-5"; if not, use "card col-3 container d-flex mb-5"
  <div
    className={
      props.mobile ? "card d-flex mt-5" : "card col-3 container d-flex mb-5"
    }
  >
    <div className="card-body container">
      <div className="mt-3">
        <p>
          <b>0/5</b>
          books checked out
        </p>
        <hr />
        {/* if we have book and its number of copies available which is positive, we let it "avaialble"; if not let it Wait List */}
        {props.book &&
          props.book.copiesAvailable &&
          props.book.copiesAvailable > 0 ? (
            <h4 className="text-success">Available</h4>
          ) : (
            <h4 className="text-danger">Wait List</h4>
          )
        }
      </div>
    </div>
  </div>
)
```

In “return” statement, then create “copies” and “available” part.

```
/* text for copies and copies available */


<p className="col-6 lead">
    <b>{props.book?.copies}</b>
    copies
  </p>
  <p className="col-6 lead">
    <b>{props.book?.copiesAvailable}</b>
    available
  </p>
</div>
</div>


```

In “return” statement, create “sign in ” part.

```
/* sign in button */
<Link to="/#" className="btn btn-success btn-lg">
  Sign In
</Link>
<hr />
<p className="mt-3">
  This number can change until placing order has been complete.
</p>
<p>Sign in to be able to leave a review.</p>
</div>
</div>
```

Go to BookCheckoutPage.tsx, import “CheckoutAndReviewBox” component in the part of code for the desktop version, and set mobile={false} in its properties.

```
layouts > BookCheckOutPage > BookCheckOutPage.tsx > BookCheckOutPage
57
58
59   return (
60     <div>
61       <div className="container d-none d-lg-block">
62         <div className="row mt-5">
63           /* left side of the page: book image part1 */
64           <div className="col-sm-2 col-md-2">
65             {book?.img ? (
66               <img src={book?.img} width="226" height="349" alt="Book" />
67             ) : (
68               <img
69                 src={require("../../Images/BooksImages/book-luv2code-1000.png")}
70                 width="226"
71                 height="349"
72                 alt="Book"
73               />
74             )}
75           </div>
76           /* right side of the page: book info part */
77           <div className="col-4 col-md-4 container">
78             <div className="ml-2">
79               <h2>{book?.title}</h2>
80               <h5 className="text-primary">{book?.author}</h5>
81               <p className="lead">{book?.description}</p>
82               /* import StarsReview Component */
83               <StarsReview rating={2.5} size={32} />
84             </div>
85           </div>
86           <CheckoutAndReviewBox book={book} mobile={[false]} />
87         </div>
88       </div>
```

Scroll down the page , Go to the part of code for the mobile, import “CheckoutAndReviewBox” component, and set mobile={true} in its properties.

```
105  /* book info part */
106  <div className="mt-4">
107    <div className="ml-2">
108      <h2>{book?.title}</h2>
109      <h5 className="text-primary">{book?.author}</h5>
110      <p className="lead">{book?.description}</p>
111      {/* import StarsReview Component */}
112      <StarsReview rating={2.5} size={32} />
113    </div>
114  </div>
115  <CheckoutAndReviewBox book={book} mobile={true} />
116  <hr />
117</div>
118</div>
119);
120};
```

Go to <http://localhost:3000/checkout/2>, we can see the golden star review and checkout and review box.



Mobile Version. We can see the checkout and review box still work.

