

UNREAD BOOK APP

Kelsie, Delphine, Andromeda & Anne



MVP

A user should be able to:

- Search books from an API, find a specific one and add it to your database
- View and delete it.
- Mark the book as read, or unread.

Extension

- Adding a user

Learning goals

Kelsie

More experience in handling complex APIs, and working as a group!

Andromeda

Better understanding of state and props, working with APIs and practice more styling

Delphine

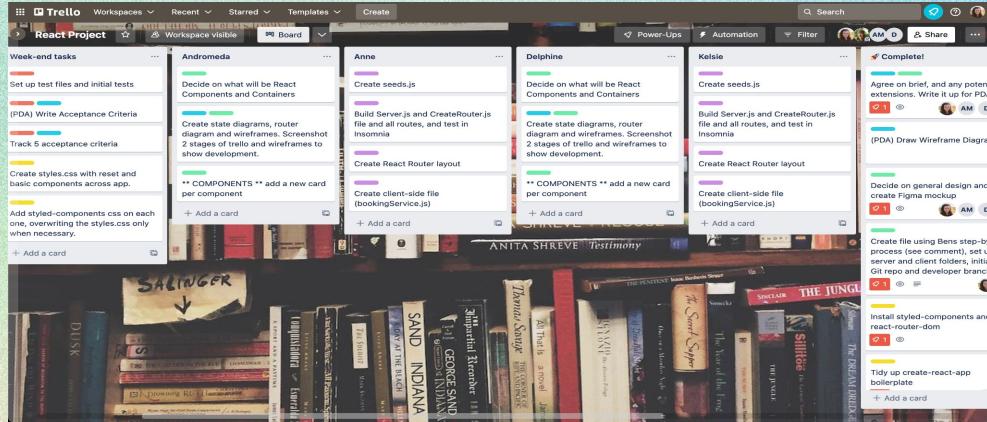
More practise with a non-relational document based database

Anne

Confidence with state and props and using an API.

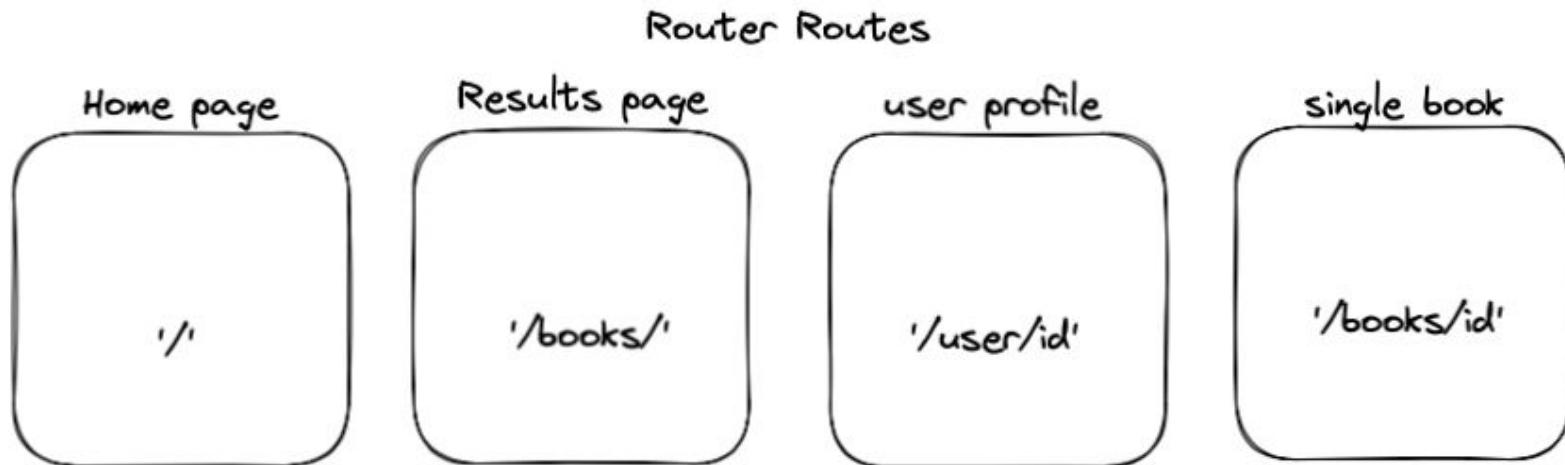
Trello

How it started...

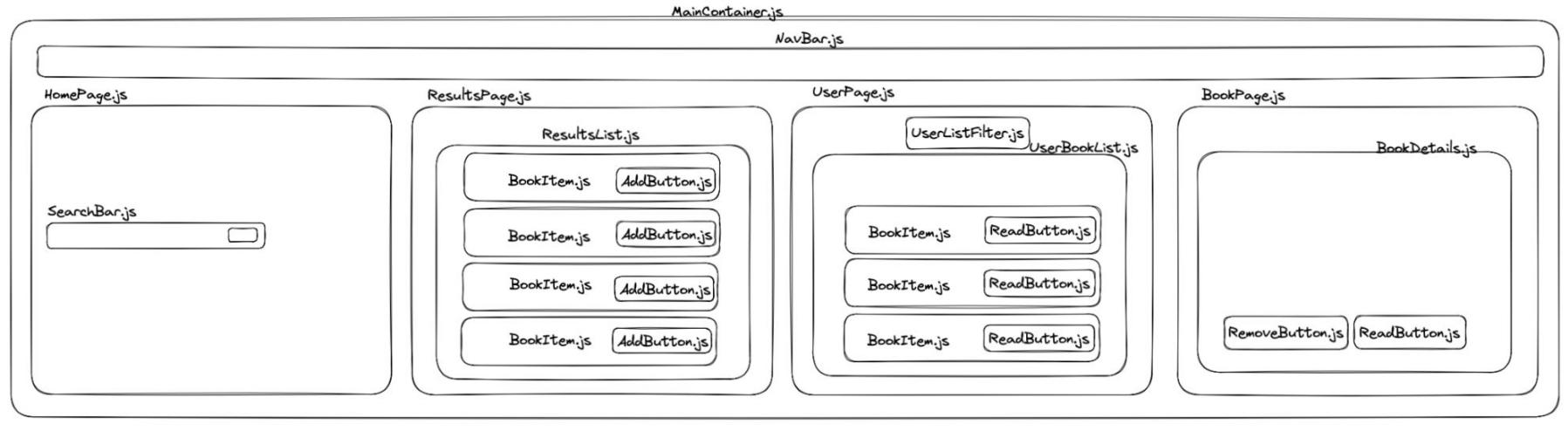


... how it ended

Planning using exalidraw



Planning using Excalidraw



Planning using Excalidraw

Main Container

State: [searchResults, setSearchResults] = useState([])
State: [userBookList, setUserBookList] = useState([])

Navbar

Homepage

SearchBar

State: [searchbarInput, setSearchbarInput] = useState("")
Props: {handleSubmitForm(searchbarValue)}

ResultsPage

Props: {searchResults, selectedBooks, onBookSelected}

AddButton

Props: {handleAddBook(book)}

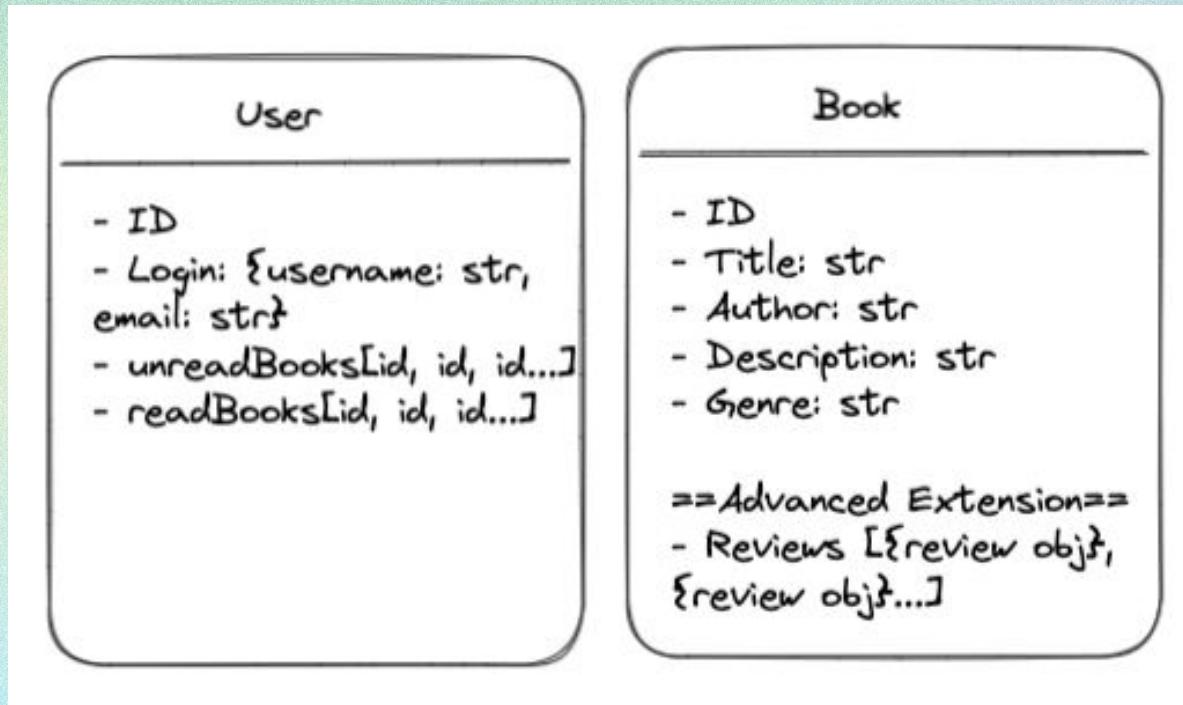
UserPage

State: [filterAsUnread, setFilterAsUnread] = useState(true)
Props: {userBookList}

BookPage

Props: {book, handleRemove(id), handleRead(id)}

Class Diagrams



Planning using Figma

```
--clr-primary-025: ■#F6FEF9;  
--clr-primary-050: ■#EDFCF2;  
--clr-primary-100: ■#D3F8DF;  
--clr-primary-200: ■#AAF0C4;  
--clr-primary-300: ■#73E2A3;  
--clr-primary-400: ■#3CCB7F;  
--clr-primary-500: ■#16B364;  
--clr-primary-600: ■#099250;  
--clr-primary-700: ■#087443;  
--clr-primary-800: ■#095C37;  
--clr-primary-900: ■#084C2E;
```

```
--clr-grey-025: ■#FCFCFD;  
--clr-grey-050: ■#F9FAFB;  
--clr-grey-100: ■#F2F4F7;  
--clr-grey-200: ■#EAECF0;  
--clr-grey-300: ■#D0D5DD;  
--clr-grey-400: ■#98A2B3;  
--clr-grey-500: ■#667085;  
--clr-grey-600: ■#475467;  
--clr-grey-700: ■#344054;  
--clr-grey-800: ■#1D2939;  
--clr-grey-900: ■#101828;
```

The image displays two screenshots of a digital book reading application. The left screenshot shows the 'Home' screen, which features a search bar at the top, followed by a section titled 'Search the collection' with a placeholder 'Browse the collection and add all the books that you've read to your reading list!'. Below this is a grid of various book covers, including 'The Lord of the Rings', 'Lord of the Flies' by William Golding, 'Animal Farm' by George Orwell, 'The Great Gatsby' by Scott Fitzgerald, and 'The Gruffalo' by Julia Donaldson. The right screenshot shows the 'Results' screen, which has a similar header and layout. It displays a title '16 Search Results' and a sub-header 'Decide what to read next'. Below this, there are four identical card-like results for 'The Lord of the Rings' by J.R.R. Tolkien. Each result card includes a small thumbnail of the book cover, the author's name, the title, a brief description stating it's the first of three volumes, and genre tags 'Crime' and 'Romance' with a 'Mark as Read' button.

Code we are particularly proud of

ALL OF IT

Code we are particularly proud of

```
useEffect( () => {
  fetch(`https://openlibrary.org/search.json?q=${searchBarInput}`)
    .then(res => res.json())
    .then(bookData => {
      const bookPromises = bookData.docs.splice(0, 20).map((doc) => {
        return fetch(`https://openlibrary.org/${doc.key}.json`)
          | .then(res => res.json())
      });
      Promise.all(bookPromises)
        .then(books => {
          setSearchResults(books);
          setIsLoading(false)
        });
    });
}, [searchBarInput]);
```

```
const onBookRemoved = (bookToRemove) => {
  const copyUser = {...user}
  console.log('bookToRemove', bookToRemove);
  const bookToRemoveIndex = copyUser.unreadBooks.findIndex(book => book._id === bookToRemove._id)
  console.log('bookToRemoveIndex', bookToRemoveIndex);
  if (bookToRemoveIndex >= 0) {
    copyUser.unreadBooks.splice(bookToRemoveIndex, 1)
  }
  else {
    console.log('else triggered');
    const bookToRemoveIndex = copyUser.readBooks.findIndex(book => book._id === bookToRemove._id)
    copyUser.readBooks.splice(bookToRemoveIndex, 1)
  }
  BookService.deleteBook(bookToRemove._id);
  setUser(copyUser)
  UserService.updateUser(copyUser)
}
```

```
const UserBookList = ({user, readState, onBookRemoved, onBookRead, onBookUnread}) => {
  const unreadUserBooksNodes = user.unreadBooks.map((userBook, index) => {
    | | return <li key={index}><UserBookItem readState={readState} userBook={userBook} onBookRemoved={onBookRemoved} onBookRead={onBookRead} onBookUnread={onBookUnread}/></li>
  })
  const readUserBooksNodes = user.readBooks.map((userBook, index) => {
    | | return <li key={index}><UserBookItem readState={readState} userBook={userBook} onBookRemoved={onBookRemoved} onBookRead={onBookRead} onBookUnread={onBookUnread}/></li>
  })
}
```

Our thoughts on working in a group

Two stars (and an extra star just for fun):

- Good learning from others workflow and tools they like to use
- We can do so much more in a group!
- Greater creativity! Loads of ideas on Friday!

And a wish (and an extra wish just for fun):

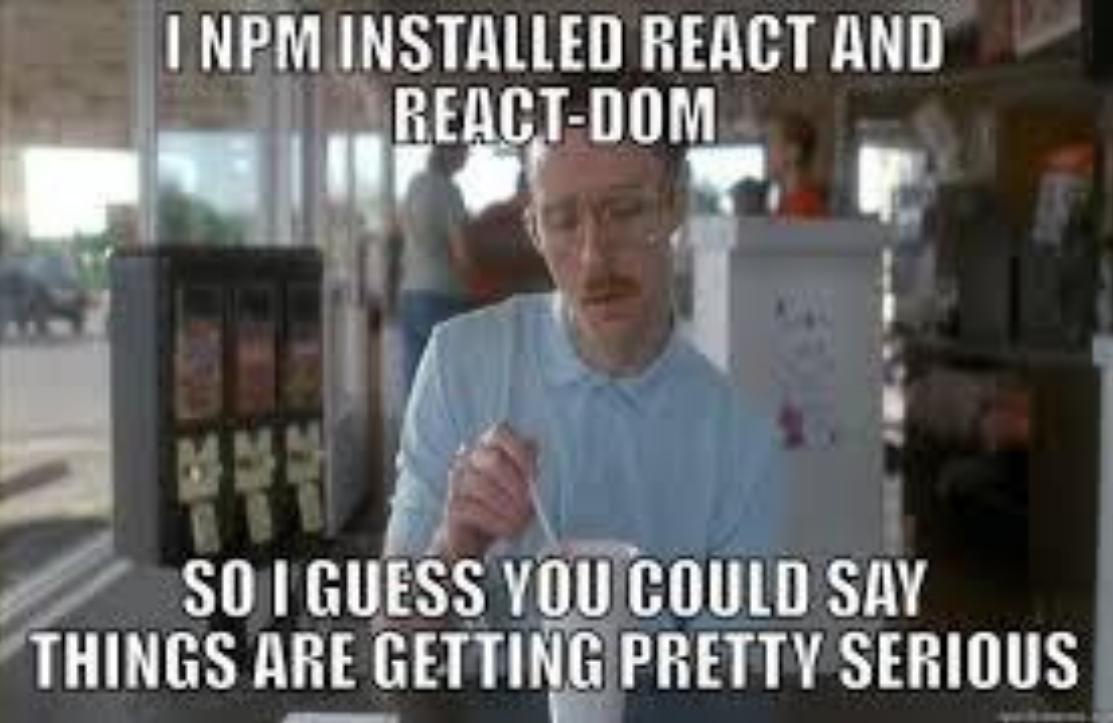
- Tricky working with people schedules and priorities at weekends
- Understanding people's code before stepping in takes time...

Show the app already



Merci

σας ευχαριστώ



I NPM INSTALLED REACT AND
REACT-DOM

SO I GUESS YOU COULD SAY
THINGS ARE GETTING PRETTY SERIOUS