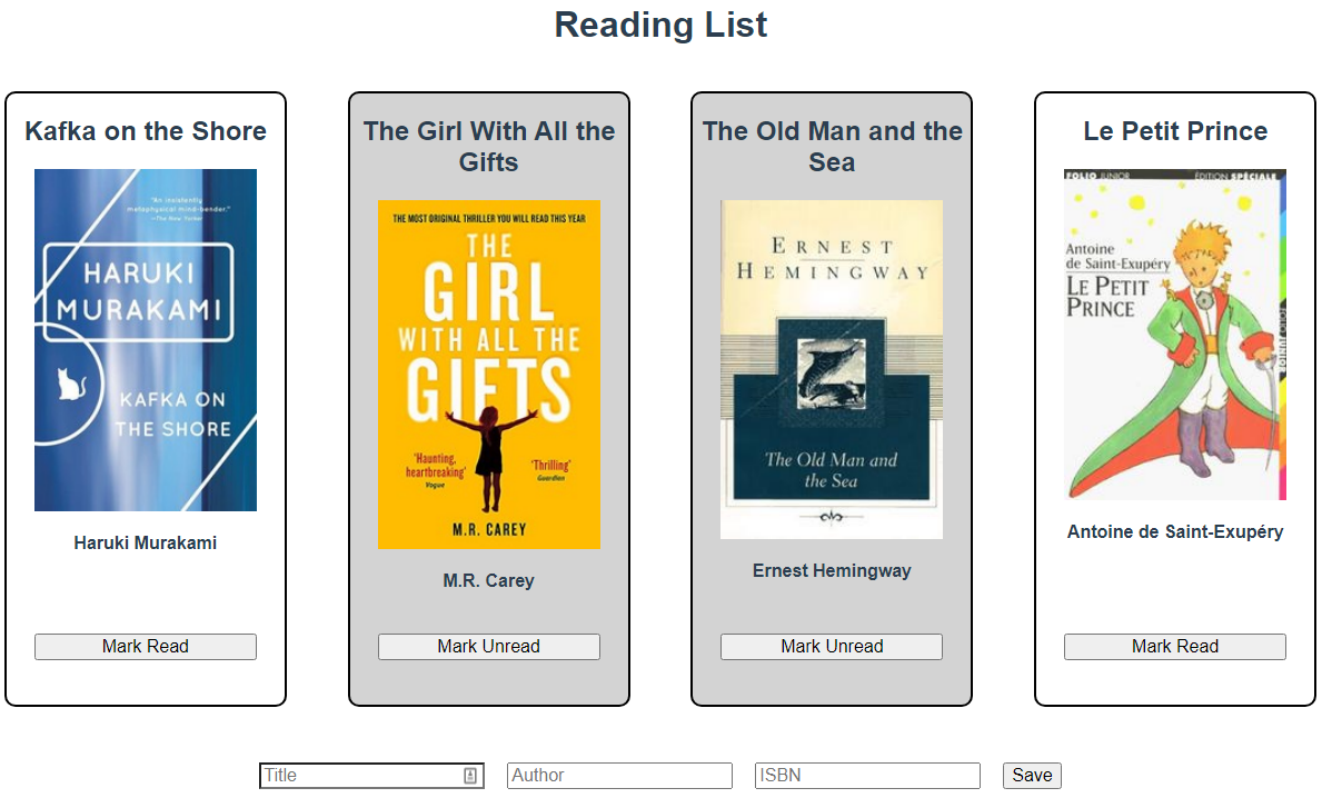
Component Communication Exercise

In this exercise, you'll take an existing component and turn it into a group of components that communicate through a Vuex datastore. Throughout this exercise, you'll move data into a Vuex datastore, set up mutations, and create new components to interact with the data.

Before you begin the exercise, run `npm install` to install any dependencies. You can run the end-to-end tests with `npm run test:e2e` or `npm run test:e2e-headless`.

After you complete the exercise steps, the final application looks similar to this:



Step One: Show reading list

First, take the following state and put it into the Vuex datastore located in `src/store/index.js`:

```
books: [
  {
    title: "Kafka on the Shore",
    author: "Haruki Murakami",
    read: false,
    isbn: "9781784877989"
  },
  {
    title: "The Girl With All the Gifts",
    author: "M.R. Carey",
    read: true,
    isbn: "9780356500157"
```

```
  },
  {
    title: "The Old Man and the Sea",
    author: "Ernest Hemingway",
    read: true,
    isbn: "9780684830490"
  },
  {
    title: "Le Petit Prince",
    author: "Antoine de Saint-Exupéry",
    read: false,
    isbn: "9783125971400"
  }
]
```

Implement the `ReadingList` component to take that data and create a `BookCard` component for each book. At this stage, you only need to create a `BookCard` component per book—it'll only show an empty box for each book until you fill in the details in the next step.

After you complete step one, run the end-to-end tests.

All tests under "Step One Tests" now pass.

Step Two: Display book details

Set up the `BookCard` component to have a prop that takes a book and shows the details of that book in its UI: title, author, and cover image.

Use the following HTML elements and CSS classes for the book details:

Book detail	HTML element	CSS class
Title	`h2`	`book-title`
Author	`h3`	`book-author`
Cover image	`img`	`book-image`

For the book's cover image, use the following for the `v-bind:src`:

```
v-bind:src="'http://covers.openlibrary.org/b/isbn/' + book.isbn + '-M.jpg'"
```

Once you complete this step, all the tests under "Step Two Tests" pass.

Step Three: Add a read/unread toggle button

Add a new control to the `BookCard` that's a button to toggle the read and unread status of a book. The button must indicate what clicking the button sets the status to. All books with a read status must have the `read` class added to the card.

Remember that the read status must be set through a mutation, not set directly from the `BookCard` component.

The button that sets the status to read must have a `mark-read` class, and the button that sets the status to unread must have a `mark-unread` class.

Once you complete this step, all tests under "Step Three Tests" pass.

Step Four: Add a `NewBookForm` component

Create a new component that has a form for a user to add a new book to the list. The elements must have the following classes:

Element	CSS class
Form	`new-book-form`
Title Input Field	`title-input`
Author Input Field	`author-input`
ISBN Input Field	`isbn-input`

Lastly, add the component to the page in the `App` component.

Now, all tests pass.