**Home**

The Home screen is the first page the user sees. It is displayed at “ / ”

The Home screen has the following features:

* The path to this screen should be “ / ”
* A "Create Deck" button is shown and clicking it brings the user to the Create Deck screen.
* Existing decks are each shown with the deck name, the number of cards, and a “Study,” “View,” and “Delete” button.
* Clicking the “Study” button brings the user to the Study screen.
* Clicking the “Edit” button brings the user to the Edit Deck screen.
* Clicking the “Delete” button shows a warning message before deleting the deck.

#### Delete Deck prompt

When the user clicks the "Delete" button, a warning message is shown and the user can click "OK" or "Cancel". If the user clicks "OK", the deck is deleted and the deleted deck is no longer visible on the Home screen.

You can use [window.confirm()](https://developer.mozilla.org/en-US/docs/Web/API/Window/confirm" \t "_blank) to create the modal dialog shown in the screenshot below.

### Study

The Study screen is displayed at /decks/:deckId/study.

The Study screen has the following features:

* The path to this screen should include the *deckId* (i.e., /decks/:deckId/study).
* You must use the readDeck() function from src/utils/api/index.js to load the deck that is being studied.
* There is a breadcrumb navigation bar with links to home /, followed by the name of the deck being studied and finally the text Study (e.g., Home/Rendering In React/Study).
* The deck title (i.e., "Study: Rendering in React" ) is shown on the screen.
* Cards are shown one at a time, front-side first.
* A button at the bottom of each card "flips" it to the other side.
* After flipping the card, the screen shows a next button (see the "Next button" section below) to continue to the next card.
* After the final card in the deck has been shown, a message (see the "Restart prompt" section below) is shown offering the user the opportunity to restart the deck.
  + If the user does not restart the deck, they should return to the home screen.
* Studying a deck with two or fewer cards should display a "Not enough cards" message (see the "Not enough cards" section below) and a button to add cards to the deck.

#### Next button

The next button appears after the card is flipped.

#### Restart prompt

When all cards are finished, a message is shown and the user is offered the opportunity to restart the deck. If the user does not restart the deck, they return to the home screen.

You can use [window.confirm()](https://developer.mozilla.org/en-US/docs/Web/API/Window/confirm" \t "_blank) to create the modal dialog shown in the screenshot below.

#### Not enough cards

Studying a Deck with two or fewer cards should display a "Not enough cards" message and a button to add cards to the deck.

Clicking the "Add Cards" button should take the user to the Add Card screen.

### Create Deck

The Home screen has a "Create Deck" button that brings the user to the Create Deck screen.

The Create Deck screen has the following features:

* The path to this screen should be /decks/new.
* There is a breadcrumb navigation bar with a link to home / followed by the text Create Deck (i.e., Home/Create Deck).
* A form is shown with the appropriate fields for creating a new deck.
  + The name field is an <input> field of type text.
  + The description field is a <textarea> field that can be multiple lines of text.
* If the user clicks "submit", the user is taken to the Deck screen.
* If the user clicks "cancel", the user is taken to the Home screen.

## Deck

The Deck screen displays all of the information about a deck.

The Deck screen has the following features:

* The path to this screen should include the *deckId* (i.e., /decks/:deckId).
* You must use the readDeck() function from src/utils/api/index.js to load the existing deck.
* There is a breadcrumb navigation bar with a link to home / followed by the name of the deck (e.g., Home/React Router).
* The screen includes the deck name (e.g., "React Router") and deck description (e.g., "React Router is a collection of navigational components that compose declaratively in your application").
* The screen includes "Edit", "Study", "Add Cards", and "Delete" buttons. Each button takes the user to a different destination, as follows:

| Button Clicked | Destination |  
| -------------- | ---------------------------------------------------------------------------------------------- |  
| "Edit" | Edit Deck Screen |  
| "Study" | Study screen |  
| "Add Cards" | Add Card screen |  
| "Delete" | Shows a warning message before deleting the deck]( See the "Delete Card Prompt" section below) |

* Each card in the deck:
  + is listed on the page under the "Cards" heading.
  + shows a question and the answer to the question.
  + has an “Edit” button that takes the user to the Edit Card screen when clicked.
  + has a “Delete” button that allows that card to be deleted.

### Delete Card Prompt

When the user clicks the "Delete" button associated with a card, a warning message is shown and the user can click "OK" or "Cancel". If the user clicks "OK", the card is deleted.

You can use [window.confirm()](https://developer.mozilla.org/en-US/docs/Web/API/Window/confirm" \t "_blank) to create the modal dialog shown in the screenshot below.

### Edit Deck

The Edit Deck screen allows the user to modify information on an existing deck.

The Edit Deck screen has the following features:

* The path to this screen should include the *deckId*(i.e., /decks/:deckId/edit).
* You must use the readDeck() function from src/utils/api/index.js to load the existing deck.
* There is a breadcrumb navigation bar with a link to home /, followed by the name of the deck being edited, and finally the text Edit Deck (e.g., Home/Rendering in React/Edit Deck).
* It displays the same form as the Create Deck screen, except it is pre-filled with information for the existing deck.
* The user can edit and update the form.
* If the user clicks "Cancel", the user is taken to the Deck screen.

### Add Card

The Add Card screen allows the user to add a new card to an existing deck.

The Add Card screen has the following features:

* The path to this screen should include the *deckId* (i.e., /decks/:deckId/cards/new).
* You must use the readDeck() function from src/utils/api/index.js to load the deck that you're adding the card to.
* There is a breadcrumb navigation bar with a link to home /, followed by the name of the deck to which the cards are being added, and finally the text Add Card (e.g., Home/React Router/Add Card).
* The screen displays the "React Router: Add Card" deck title.
* A form is shown with the "front" and "back" fields for a new card. Both fields use a <textarea> tag that can accommodate multiple lines of text.
* If the user clicks "Save", a new card is created and associated with the relevant deck. Then the form is cleared and the process for adding a card is restarted.
* If the user clicks "Done", the user is taken to the Deck screen.

The Edit Card screen has the following features:

* The path to this screen should include the *deckId* and the *cardId* (i.e., /decks/:deckId/cards/:cardId/edit).
* You must use the readDeck() function from src/utils/api/index.js to load the deck that contains the card to be edited. Additionally, you must use the readCard() function from src/utils/api/index.js to load the card that you want to edit.
* There is a breadcrumb navigation bar with a link to home /, followed by the name of the deck of which the edited card is a member, and finally the text Edit Card :cardId (e.g., Home/Deck React Router/Edit Card 4).
* It displays the same form as the Add Card screen, except it is pre-filled with information for the existing card. It can be edited and updated.
* If the user clicks on either "Save" or "Cancel", the user is taken to the [Deck screen](https://www.qualified.io/assess/" \l "deck" \t "_blank).

**Note:** In addition to needing to pass the tests and requirements in the instructions here, please review the Rubric Requirements for the human-graded part of this project in your Thinkful curriculum page.