Project Overview

For the **UdaciCards** project, you will build a mobile application (Android or iOS - or both) that allows users to study collections of flashcards. The app will allow users to create different categories of flashcards called "decks", add flashcards to those decks, then take quizzes on those decks.

Why this project?

This project encompasses the fundamental aspects of building a native application including handling infinite lists, routing, and user input. By building this project, you will gain an understanding of how to use React Native to build an iOS and Android application.

Specification

You'll create your project using **create-react-native-app**. There will be no starter code that you need to download.

The specification provided below is the minimum required for this project. You may extend your project as you like, however.

Specific Requirements

- Use create-react-native-app to build your project.
- Allow users to create a deck which can hold an unlimited number of cards.
- Allow users to add a card to a specific deck.
- The front of the card should display the question.
- The back of the card should display the answer.
- Users should be able to quiz themselves on a specific deck and receive a score once they're done.
- Users should receive a notification to remind themselves to study if they haven't already for that day.

Views

Your application should have, at a minimum, five views.

- Deck List View (Default View)
  - displays the title of each Deck
  - displays the number of cards in each deck

DECKS

udacicards

3 cards

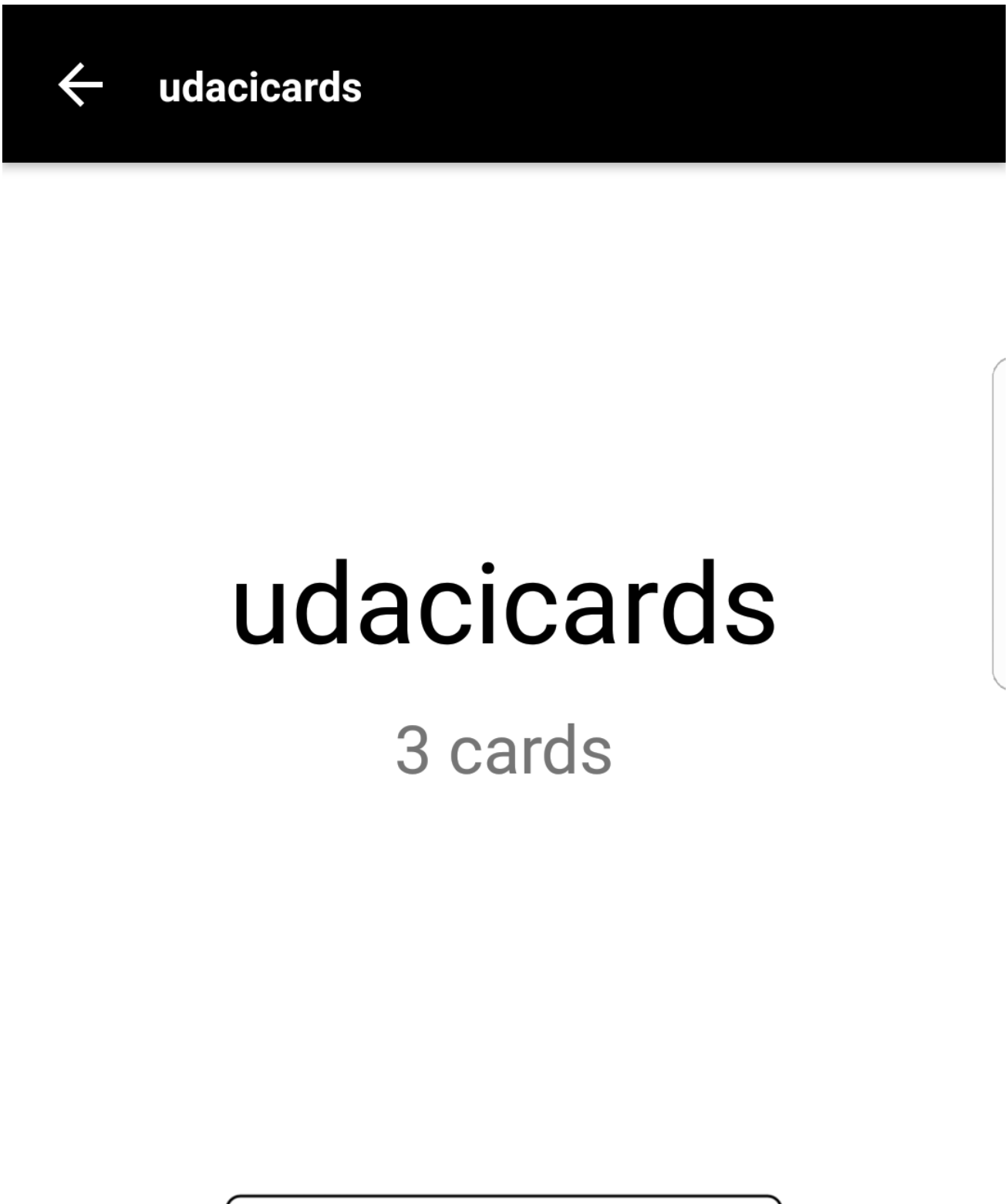
new deck

0 cards

New deck 2

0 cards

- Individual Deck View
  - displays the title of the Deck
  - displays the number of cards in the deck
  - displays an option to start a quiz on this specific deck
  - An option to add a new question to the deck



Individual Deck View

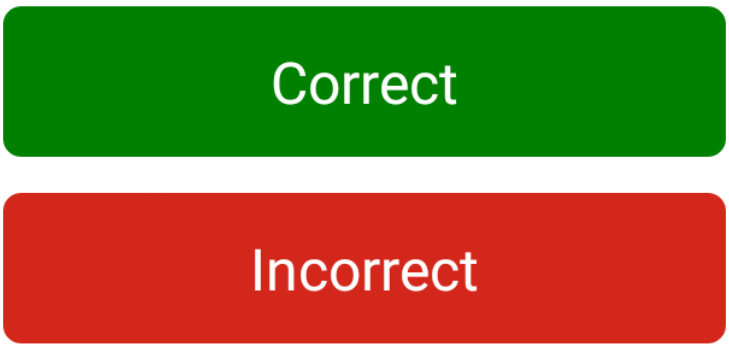
- Quiz View
  - displays a card question
  - an option to view the answer (flips the card)
  - a "Correct" button
  - an "Incorrect" button
  - the number of cards left in the quiz
  - Displays the percentage correct once the quiz is complete

2 / 2

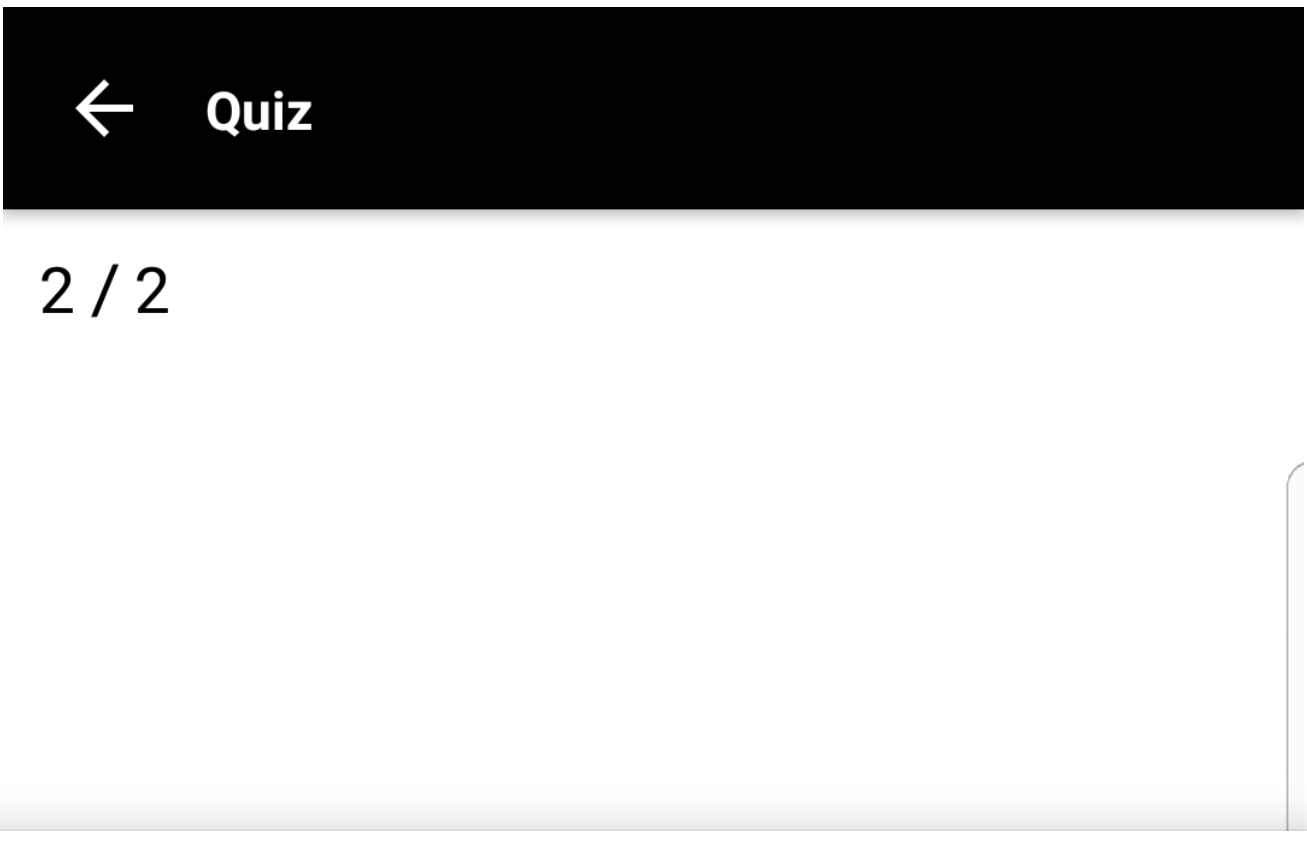
Does React Native work with

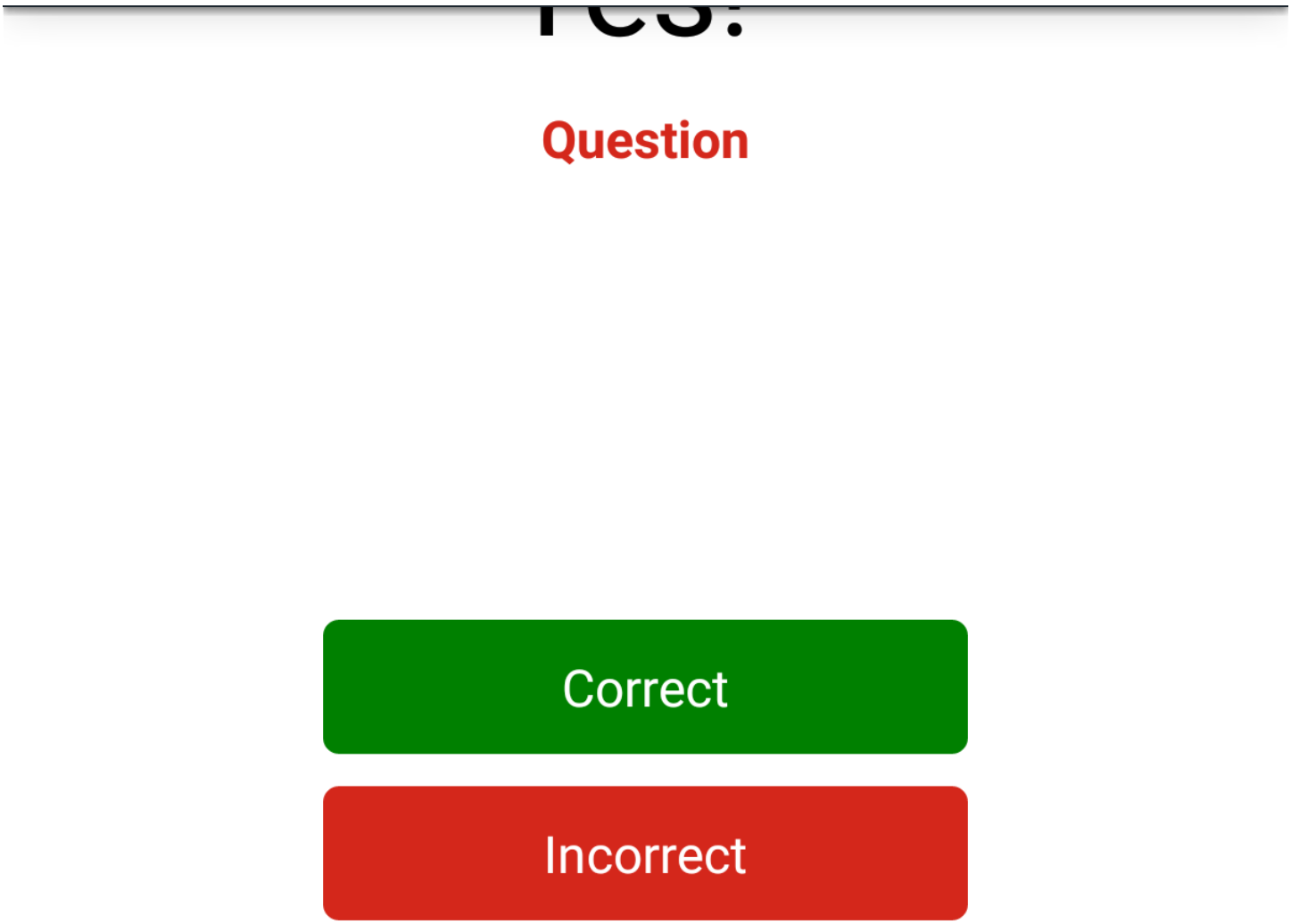
Android?

Answer

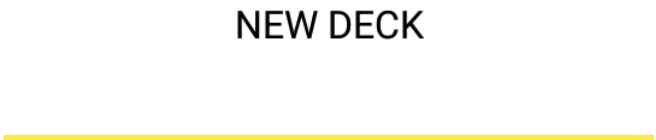


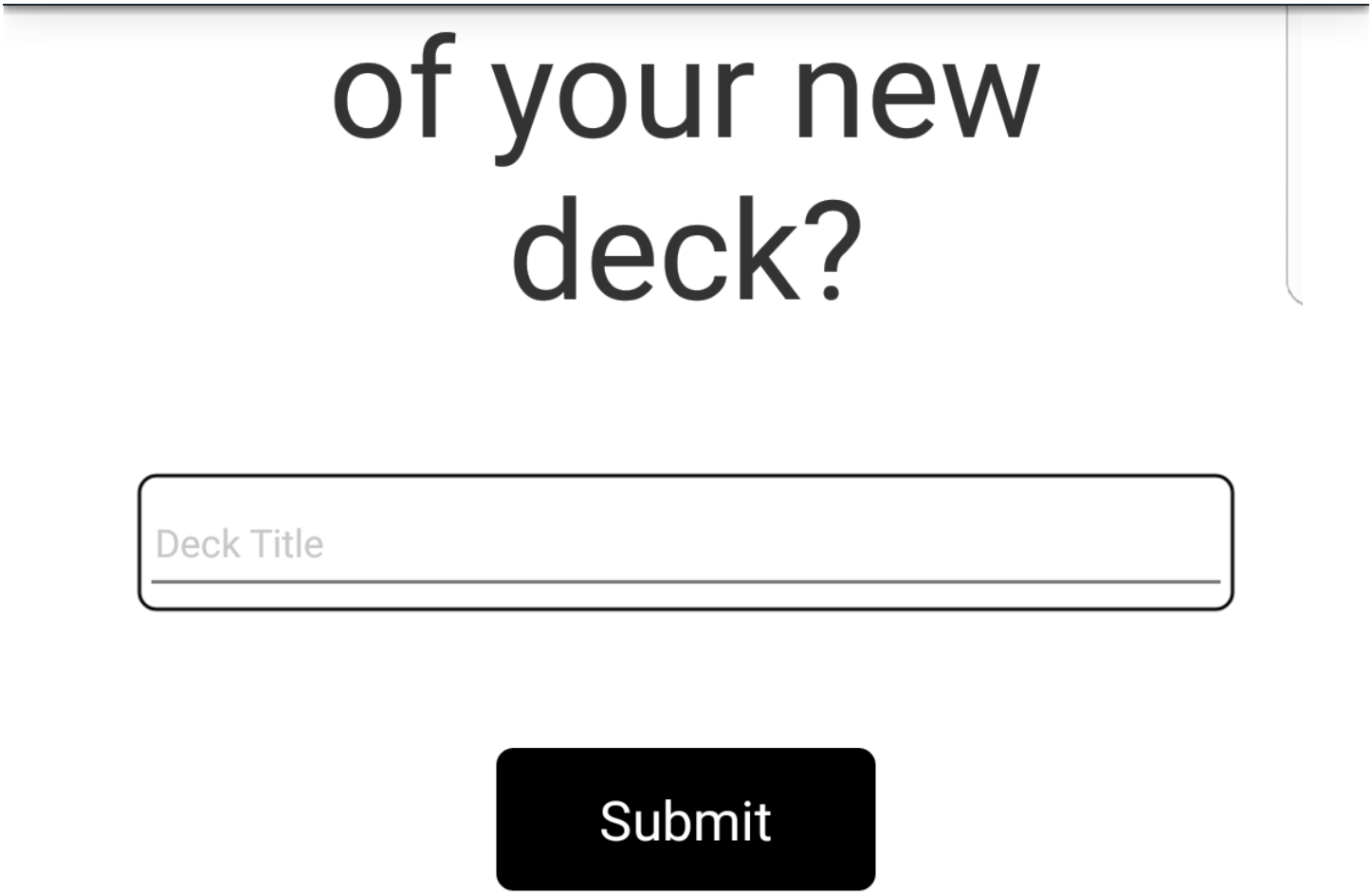
Quiz View



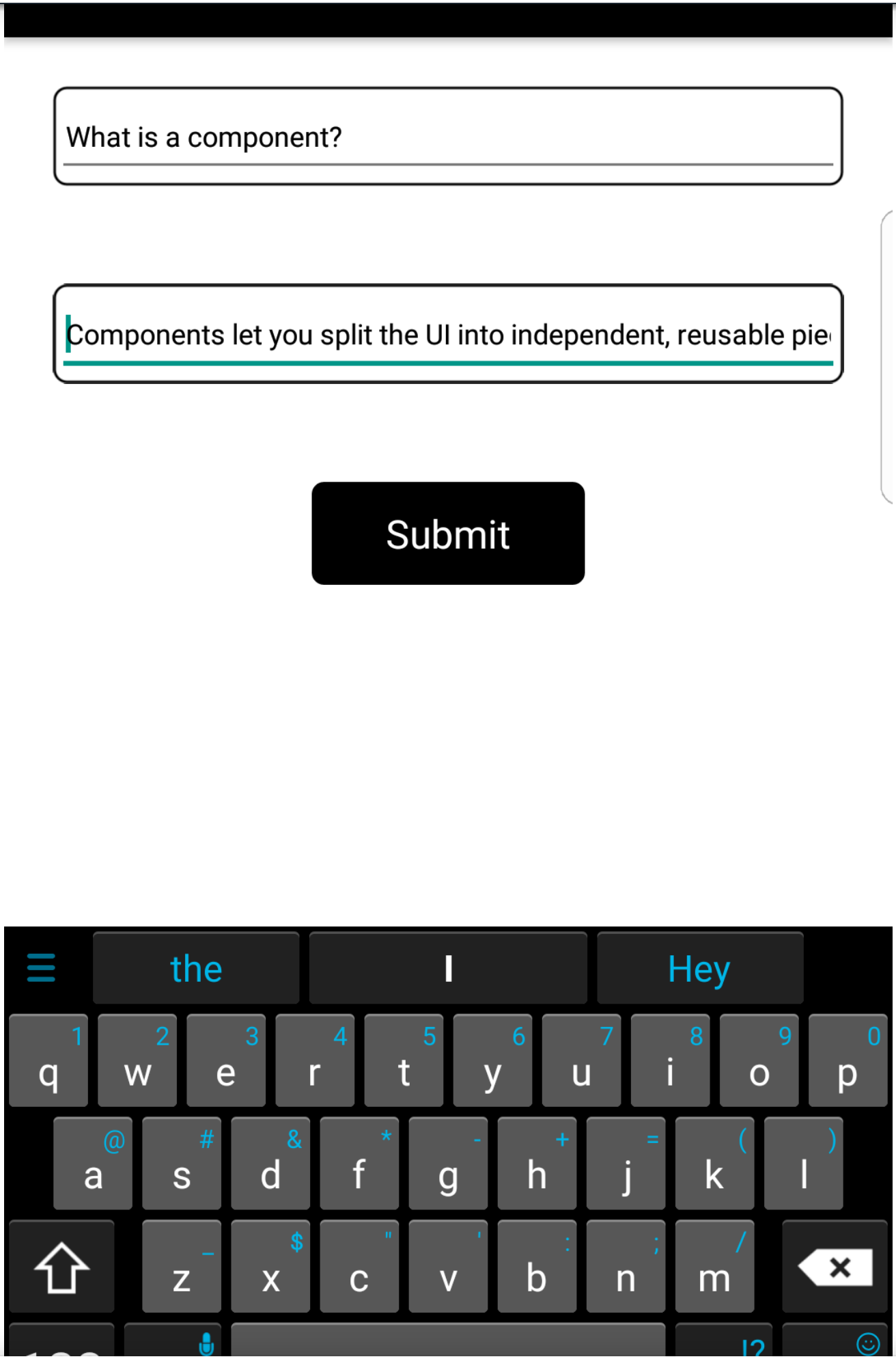


- New Deck View
  - An option to enter in the title for the new deck
  - An option to submit the new deck title





- New Question View
  - An option to enter in the question
  - An option to enter in the answer
  - An option to submit the new question


What is a component?
Components let you split the UI into independent, reusable pie
Submit
the
I
Hey

Data

We'll use `AsyncStorage` to store our decks and flashcards. Redux is optional for this project.

Using `AsyncStorage` you'll manage an object whose shape is similar to this:

```
{
  React: {
    title: 'React',
    questions: [
      {
        question: 'What is React?',
        answer: 'A library for managing user interfaces'
      },
      {
        question: 'Where do you make Ajax requests in React?',
        answer: 'The componentDidMount lifecycle event'
      }
    ]
  },
  JavaScript: {
    title: 'JavaScript',
    questions: [
      {
        question: 'What is a closure?',
        answer: 'The combination of a function and the lexical environ
      }
    ]
  }
}
```

Notice each deck creates a new key on the object. Each deck has a `title` and a `questions` key. `title` is the title for the specific deck and `questions` is an array of questions and answers for that deck.

Tip

`getDecks`: return all of the decks along with their titles, questions, and answers.
`getDeck`: take in a single `id` argument and return the deck associated with that id.
`saveDeckTitle`: take in a single `title` argument and add it to the decks.
`addCardToDeck`: take in two arguments, `title` and `card`, and will add the card to the list of questions for the deck with the associated title.

NEXT