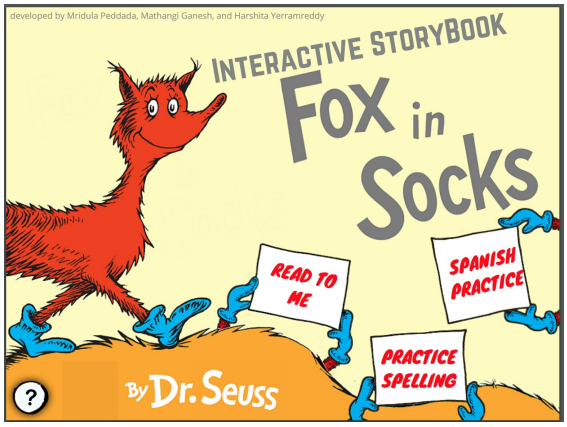
Harshita Yerramreddy
Mridula Peddada
Mathangi Ganesh
CS 230 Final Project

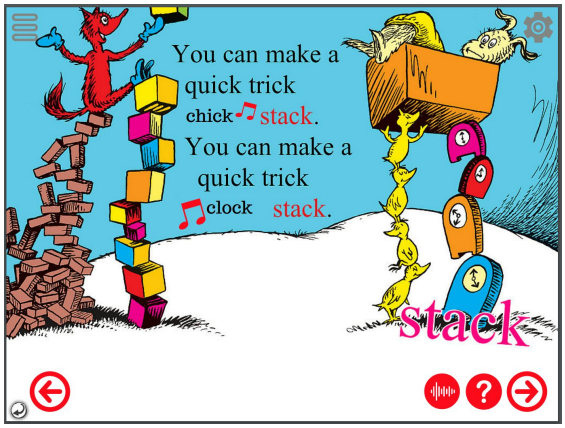
Phase 1:

A. User Manual

Welcome to our interactive storybook! This is a children's app that allows the user to read a Dr. Seuss book, such as Fox in Socks, with added effects and features.



This is the home page GUI. This panel includes a large image of the book cover, with three buttons for the three storybook modes: interactive reader, spelling practice, and language (Spanish) practice). Clicking on either button will lead the user to the first book page with the features of that mode. The '?' button will give the user instructions on how to use the program.

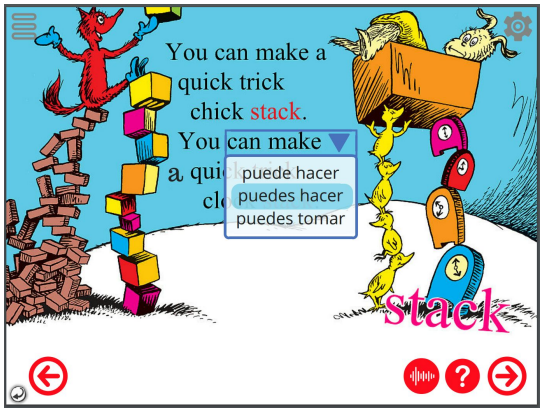


This is the first storybook mode: interactive reader. In every storybook mode, there is a set of buttons on the four corners of the window to help navigate the app and page. The three horizontal lines on the top left-hand corner is a menu button, and will allow the user to go back to the home page, or to a specific page in the book. The gear button on the right-hand side is for the app settings, which will allow the user to switch between storybook modes for that current page and all pages thereafter. On the bottom, the left arrow is to go to the previous page, and the right arrow is to go to the next page. The red 'play' button to the left of the '?' allows the user to hear a recording of the text on that page.

In this storybook mode, the simplest of the three, the user can press the 'play' button to hear a recording of the book. The user can also click on elements of the page, such as certain words with a music symbol or images, to hear a sound effect associated with that thing. Here, for example, the 'chick' text will make a chirping noise when clicked.

In the ‘Practice Spelling’ storybook mode, the user can fill-in-the-blank the appropriate spelling for words that are removed from the sentence. The sentence will automatically be played out-loud, and the user can fill in the blank spaces with the appropriate word. The user can re-play the audio recording multiple times as need to complete the exercise. If the proper spelling is entered, a sound such as ‘Yay!’ or ‘Woohoo!’ will be played to congratulate the user. If the spelling is incorrect, a sound such as ‘Aww’ will be played, and a message will be displayed to try again, or to show the answer.

In the third storybook mode, ‘Spanish Practice,’ the user can practice translating words from English to Spanish by using dropdown menus. On each page, there will be one or two words surrounded by a drop-down box, from which the user can select the correct Spanish translation. Again, if the correct answer is chosen, the appropriate sound effect will be played; otherwise, the user will be told to try again, and the combo box will be reset.



B. Technical report

ADTs that will be used:

1. **Hashtable:** Our storybook includes a spelling game that requires users to type in the correct spelling of words. We have two activities that will incorporate a spell check mechanism:

    a. The user hears an audio recording of the word and has to type in the correct spelling
    b. The user has to correct a misspelled word

    To check if the user spelled the word correctly, we will use a hashtable. Similar to the Dictionary class that we implemented in lab, we will use the hashtable to see if word the user typed in exists in a dictionary that we will have created.

2. **LinkedList:** Will store Page objects. Our book will be made up of numerous Page objects stored in a Book class, that will hold a LinkedList of Pages. We chose to use a LinkedList because it will be easier to remove and add pages to the book if necessary. Size won't be a limiting factor, and we will be able to easily retrieve Page objects (i.e. via Table of Contents or a given page number)

3. **Array:** Our storybook also has a Spanish-English translation activity, where the user will see the word written in English and will have to choose the correct corresponding Spanish word from a JComboBox. We will use an array to store the contents of the JComboBox.

Important Classes:

Page Class: Each page in our storybook will have some basic attributes such as a page number, body of text, the setting (what kind of page it is: audio-interactive, spelling activity, or translation activity).

Methods:

i. **getSettings()**
ii. **getHelp()** - It will return a File object, containing useful instructions for the user.
iii. **readPage()** -The readPage() method returns an *AudioFileFormat.* It reads out the text in the page.

ReadPage Class

Methods:

I. **getSound()-** This method returns an *AudioFileFormat.*

SpellPage Class

Methods:

I. **checkSpelling() -** the check spelling method will compare two strings, the string entered by the user and the correct string (stored in an array).The method will return correctString.equals(userInput). The return type of this method is a *boolean*.

II. **correctSpelling() -** This method will return the correct word (of type *String*) associated with the blank.

TranslatePage Class

Methods:

I. **correctTranslation()**- This method will return a *boolean*. It will compare the string the user chooses from the drop down box to the correct string (stored in an array). If the input equals the correctString then the method returns true, otherwise, it returns false.

II. **getCorrectTranslation()** - This method will return a *String*. It will return the correct string associated with the appropriate drop-down box.

Book Class: This class holds all of the pages. The class will initialize a LinkedList of Page objects.

Methods:

I. **goToPage(int pageNumber)** - The Book class will contain a goToPage(int pagenumber) method which goes to a particular page based on the integer entered as the parameter.

II. **readBook()** - The readBook() will read out the text in all the pages of the book. It will loop through every page in the LinkedList and call the readPage() method on every page.

III. **nextPage()** - The method returns a *Page* object. It will return the page at index (i+1) or in other words, it will basically return the next page of the Book.

IV. **previousPage()** - The method returns a *Page* object. It will return the page at index (i-1) or in other words, it will basically return the previous page of the Book.