React Native 2 β (3 Points)

Accessible Design

Early Due Date (+1 Point of Extra Credit) Tuesday, November 17th @ 11:59 PM CDT

Regular Due Date Sunday, November 22nd @ 11:59 PM CDT

In this assignment, you will build on your React Native 2 α assignment to explore accessibility features and assistive technologies of mobile platforms. Specifically, you will integrate React Native accessibility features into your fitness tracking app to support screen reader use.

Part 1—Discovery, Planning, & Specifying: In this part, you will discover the screen reader assistive technology features of your mobile device, plan how you might support two tasks in your fitness app using these features, and develop specifications to implement these features into your RN components.

Part 2—Implementation: This part will involve implementing the specifications developed in the previous part as well as ensuring that other components do not distract a user with visual impairments.

Part 3—Testing & Demonstration: In this part, you will demonstrate the two tasks you are supporting with your implementation and capture your demonstration in the form of a narrated screen recording.

Submission Details

GitHub Classroom Starter Code

React Native 2 β will build on your implementation of React Native 2 α. You should copy your code from your React 2 α project to the React 2 β repository linked above, as that will be your starter code. When you commit/push, ensure that you are committing/pushing to the react_native2_beta repository, not react_native2_alpha. To complete the assignment, you will need to submit:

1. A completed version of this document as PDF to Canvas;
2. Your repository name and latest commit hash from GitHub Classroom E.g., “react_native2_beta-ctnelson1997, 2b0ef83”
3. A video recording of you demonstrating in MP4 format the intended use of your prototype, saved in your Google Drive folder and shared through a link (instructions) (as video files can be too large for Canvas to handle).

Part 1: Discovery, Planning, & Specifying (1.4 Point)

In this part, you will engage in discovery of the screen reader assistive technology in your mobile platform of choice, prepare tasks for supporting accessibility in your application, and design the experience for a user with visual impairment across three steps.

Step 1. Discovery of Accessibility Features (0.3 Point). In this step, you will explore the accessibility features of the mobile device platform in which you have been testing your React Native projects. Your testing environment can be an iOS or Android device using the Expo app or an iOS or Android emulator on the computer. By enabling VoiceOver in iOS[1] (Settings → Accessibility → VoiceOver) and TalkBack in Android[2] (Settings → Accessibility → TalkBack) or accessibility testing tools in your emulator (e.g., Accessibility Inspector in Xcode), you will assess how screen readers work across two applications:

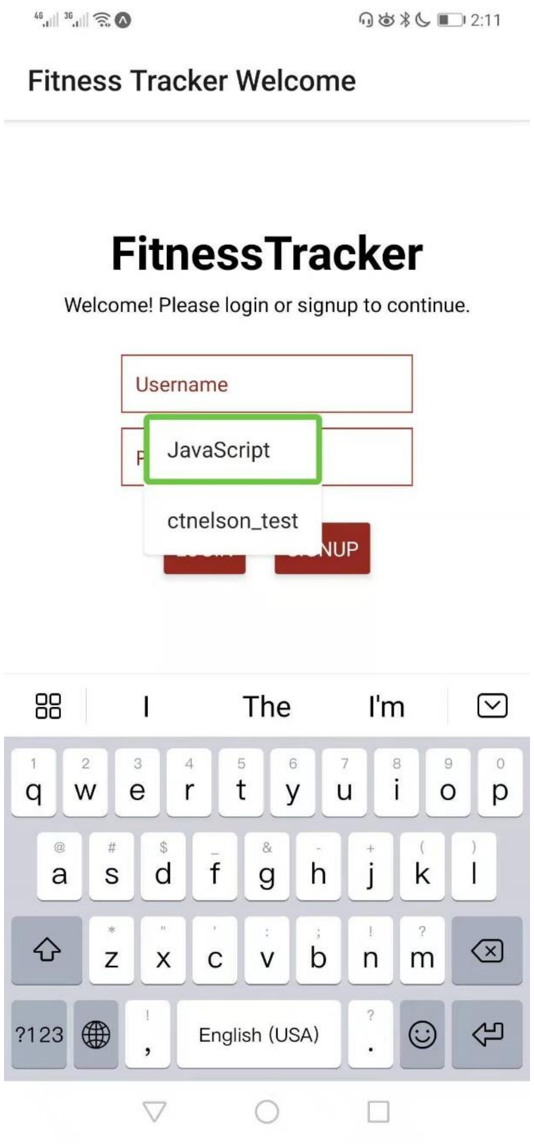
1. The latest version of your React Native fitness tracking application
2. Another application of your choice that you frequently use

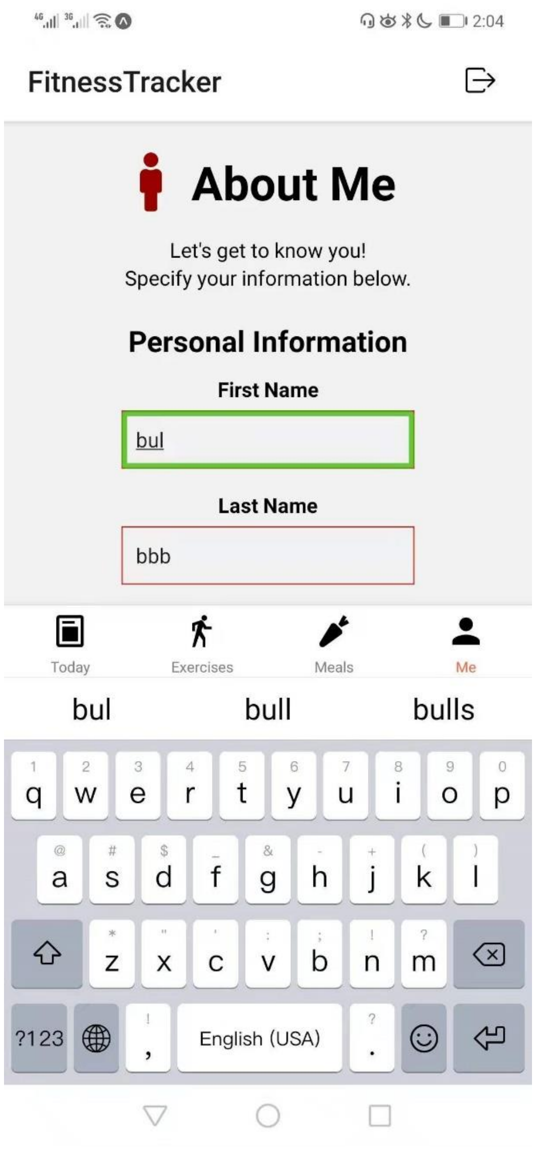
Complete or attempt to complete two common tasks in both applications with the screen reader on and report below your observations. Specifically, describe what tasks you performed or attempted to in each application and how the applications supported the task.

<task-descriptions-and-observations>

1. The latest version of your React Native fitness tracking application

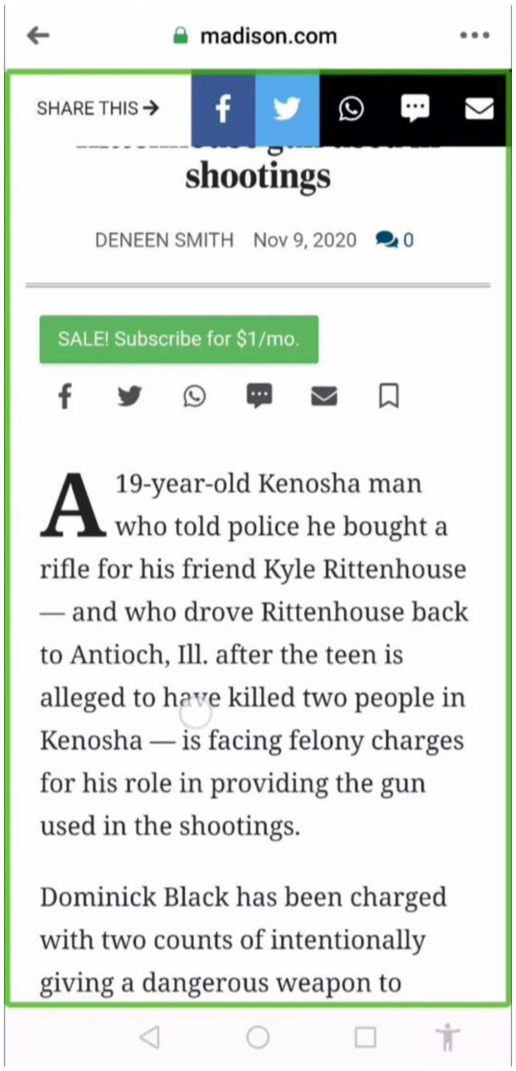
[1] You can learn more about iOS VoiceOver here: https://www.apple.com/accessibility/iphone/vision/
[2] You can learn more about Android TalkBack here: https://support.google.com/accessibility/android/answer/6283677?hl=en





First I try to log in the fitness application. I have saved the username and password in the system. Open the log in page and click the username text input, the system will start to select one and tell the user about the username. When I click the "LOGIN" button, the system guides the user verify with fingerprint with voice as well as a block to indicate the text guide.

Then I try to update the first name and last name of current user. The system will locate a text input box with a block box and speak the word in the text input. I click the input and system tells user a keyboard is open. When users tap the key, system will broadcast the character on the key to help user know what they have inputted. Also, when the user deletes a character, the system will response with voice.



madison.com

SHARE THIS →

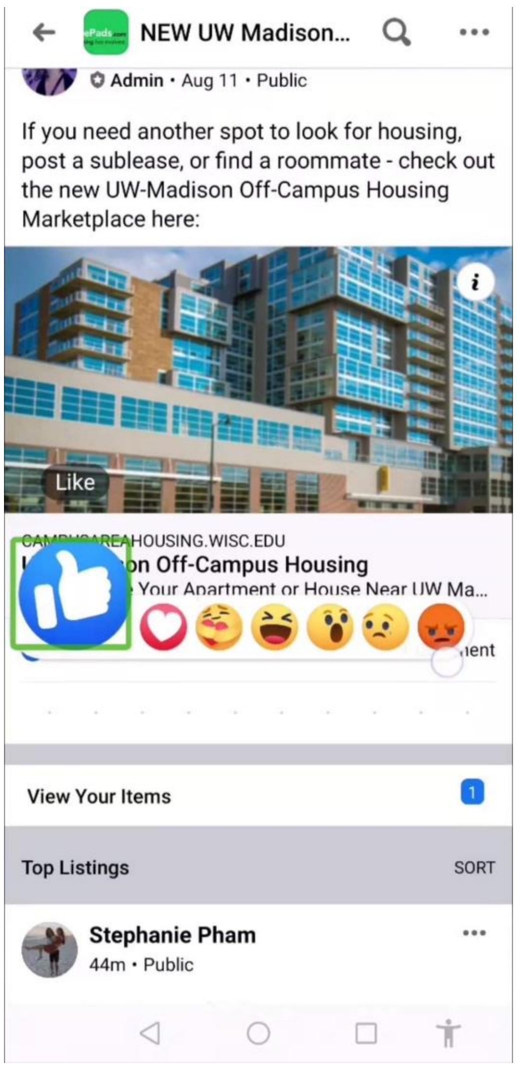
shootings

DENEEN SMITH Nov 9, 2020 0

SALE! Subscribe for $1/mo.

A 19-year-old Kenosha man who told police he bought a rifle for his friend Kyle Rittenhouse — and who drove Rittenhouse back to Antioch, Ill. after the teen is alleged to have killed two people in Kenosha — is facing felony charges for his role in providing the gun used in the shootings.

Dominick Black has been charged with two counts of intentionally giving a dangerous weapon to



NEW UW Madison...

Admin • Aug 11 • Public

If you need another spot to look for housing, post a sublease, or find a roommate - check out the new UW-Madison Off-Campus Housing Marketplace here:

CAMPUSAREAHOUSING.WISC.EDU

on Off-Campus Housing

Your Apartment or House Near UW Ma...

View Your Items 1

Top Listings SORT

Stephanie Pham

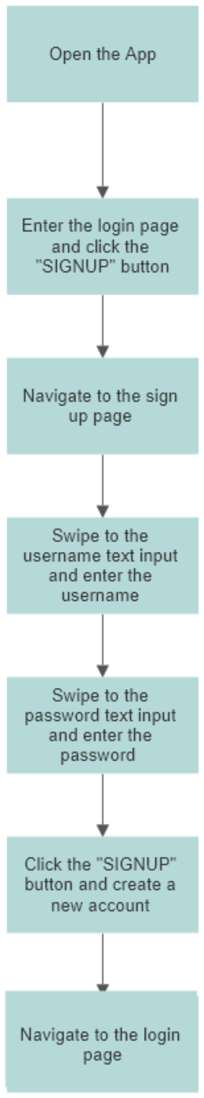
44m • Public

First I open a news page and it will display in a block box. The TalkBack read the text in the block. When the user scroll the page, a voice feedback will be provided to indicate such a performance. If the page were clicked, system will react with a touch action on the screen and voice feedback.

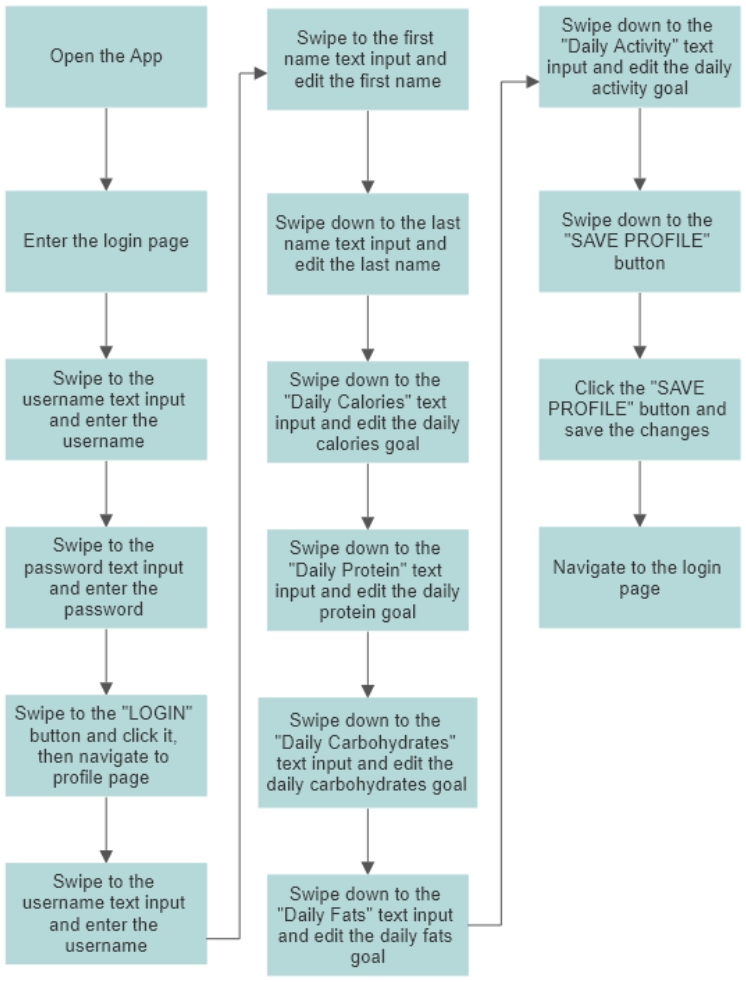
Then I try to give a thumbs-up to a posted message. Long click the like button, some memes are displayed. A block box is located at the "like" and the system gives a voice feedback "like". Switch to the right, the system gives the corresponding voice indication.

Step 2. Planning for Accessible Design (0.5 Point). In this step, you will choose <u>two</u> tasks supported by your React Native 1 α deliverable (e.g., sign up for an account) or your React Native 2 α deliverable (e.g., add an account for the current day) and map out how you expect users with blindness or severe visual impairments to interact with them given what you learned in Step 1. You can repeat the task you specified in Step 1. Specifically, you will create flowcharts of what components the user must interact with and in what order to perform the task. This activity will help you choose the right groupings for your React Native elements in order to define the accessibility features you will need. To generate flowcharts, you can use SmartDraw (using your NetID login) or free versions of other tools, such as LucidChart or Creatly.

<task-flowcharts>



Open the App
Enter the login page and click the "SIGNUP" button
Navigate to the sign up page
Swipe to the username text input and enter the username
Swipe to the password text input and enter the password
Click the "SIGNUP" button and create a new account
Navigate to the login page



Open the App
Enter the login page
Swipe to the username text input and enter the username
Swipe to the password text input and enter the password
Swipe to the "LOGIN" button and click it, then navigate to profile page
Swipe to the username text input and enter the username
Swipe to the first name text input and edit the first name
Swipe down to the last name text input and edit the last name
Swipe down to the "Daily Calories" text input and edit the daily calories goal
Swipe down to the "Daily Protein" text input and edit the daily protein goal
Swipe down to the "Daily Carbohydrates" text input and edit the daily carbohydrates goal
Swipe down to the "Daily Fats" text input and edit the daily fats goal
Swipe down to the "Daily Activity" text input and edit the daily activity goal
Swipe down to the "SAVE PROFILE" button
Click the "SAVE PROFILE" button and save the changes
Navigate to the login page

Figure 1 Figure 2

Task 1 is sign up for an account, and the flowchart is shown in Figure 1. Task 2 is profile modification, and the flowchart is shown in Figure 2.

Step 3. Specifying Accessibility Features (0.6 Point). This step will involve determining how to specify accessibility features for the components you included in your flowcharts in Step 2. For each component, you will write out how you will enable accessibility features using React Native Accessibility (review the accessibility properties), such as where accessibility features should be enabled, what labels and hints should be provided, what accessibility actions should be supported, and so on. It is important to put yourselves in the shoes of a user with visual impairments and consider how you would like to support user navigation and interaction, what the labels should say exactly so that they accurately and effectively communicate the functionality of each component.

<per-component-accessibility-specifications>

Task 1:

“SIGNUP” button in login page:
accessible={true}
accessibilityLabel={"welcome to fitness track app"}
accessibilityHint={"You will nevigate to the sign up page"}

“Username” textinput in signup page:
accessible={true}
accessibilityLabel={"Username"}
accessibilityHint={"Input a new username"}

“Password” textinput in signup page:
accessible={true}
accessibilityLabel={"Password"}
accessibilityHint={"Input password"}

“SIGNUP” button in signup page:
accessible={true}
accessibilityLabel={"Sign up successfully" + this.state.username}
accessibilityHint={"You will go back to log in page"}

Task 2:

“Username” textinput in login page:
accessible={true}
accessibilityLabel={"Username"}
accessibilityHint={"Input your username"}

“Password” textinput in login page:
accessible={true}

accessibilityLabel={"Password"}
accessibilityHint={"Input your password"}

“LOGIN” button in signup page:
accessible={true}
accessibilityLabel={"welcome back" + this.state.username}
accessibilityHint={"You will nevigate to the profile page"}

Since there will be a conflict between "placeholder" and "accessibilityLabel/Hint" for TextInput, I add the accessibility in the text component above the corresponding text input.

“First Name” text component in profile page:
accessible={true}
accessibilityLabel={"First Name"}
accessibilityHint={"Swipe down to edit your first name"}

“Last Name” text component in profile page:
accessible={true}
accessibilityLabel={"Last Name"}
accessibilityHint={"Swipe down to edit your last name"}

"Daily Calories" text component in profile page:
accessible={true}
accessibilityLabel={"Daily calories goal"}
accessibilityHint={"Swipe down to edit your daily calories goal"}

"Daily Protein" text component in profile page:
accessible={true}
accessibilityLabel={"Daily Protein goal"}
accessibilityHint={"Swipe down to edit your daily protein goal"}

"Daily Carbohydrates" text component in profile page:
accessible={true}
accessibilityLabel={"Daily Carbohydrates goal"}
accessibilityHint={"Swipe down to edit your daily carbohydrates goal"}

"Daily Fats" text component in profile page:
accessible={true}
accessibilityLabel={"Daily fat goal"}
accessibilityHint={"Swipe down to edit your daily fat goal"}

"Daily Activity" text component in profile page:
accessible={true}
accessibilityLabel={"Daily activity time goal"}
accessibilityHint={"Swipe down to edit your daily activity time goal"}

"SAVE PROFILE" button in profile page:
accessible={true}
accessibilityLabel={"Save your profile"}
accessibilityHint={"You will go back to log in page"}

Part 2: Implementation (0.8 Point)

The outcome of Steps 2 and 3 in Part 1 provides you with exact specifications for implementing the accessibility features into your code of theReact Native application. In addition to carrying out these specifications in your code, you will also have to disable accessibility features for components that do not support your tasks and might be distractions for users with visual impairments. The deliverable of this part of the assignment is the code you will submit into GitHub Classroom.

Part 3: Testing & Demonstration (0.8 Point)

In this part of the assignment, you will perform the tasks you chose in Step 2 of Part 1 with the screen reader on and capture a video of your demonstration. You can use the iOS in-built screen recorder, one of the various screen recording options on the Android, or another device (e.g., your friend's phone, or a tablet computer) to record yourself demonstrating the tasks. Save this recording into your Google Drive, set permissions such that the video is viewable for anyone with the link, and include the link in your submission on Canvas. You can save two separate video files for the two tasks, or a single video file that demonstrates the tasks back to back. In addition to capturing your screen, screen recorders can also capture your voice, and you will be asked to provide narration along with your demonstration.