CS-PRO-C83(V3)

WhiteHat Jr
Live Online Coding for Kids

TextInput AND STYLING

What is our GOAL for this MODULE?

In this class, we continued to build a Storytelling application. We styled the drawer navigation and created the form 'CreateStory.js' with a feature to preview images.

What did we ACHIEVE in the class TODAY?

- Created a CreateStory screen.
- Added a dropdown menu.
- Added a dictionary preview_images in the render() function.
- Called the setState() function which will re-render the entire screen.
- Completed the form by adding all text input fields and styling.
- Used the placeholder attribute for setting the placeholders.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- setState() function
- create the form for users
- preview images that the user selects
- text input fields and styling

How did we DO the activities?

1. Start by working on the CreateStory screen. Create a form through which the user will be able to submit a new story.

2. Use predefined images. This way, all the images in our app would align with the theme of the app.

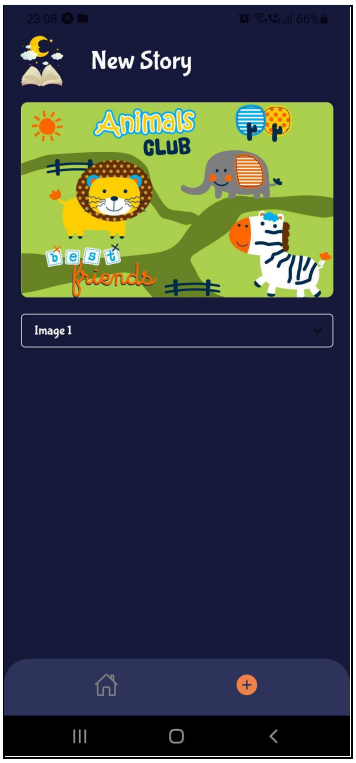
3. Add a dropdown menu. Keep an image selected by default and display the preview of that image.

```
import React, { Component } from "react";
import {
View,
Text,
StyleSheet,
SafeAreaView,
Platform,
StatusBar,
Image,
ScrollView,
TextInput,
Dimensions
} from "react-native";
import { RFValue } from "react-native-responsive-fontsize";
import DropDownPicker from "react-native-dropdown-picker";
```

4. In the render() function inside the **if-else** condition, check if fonts are loaded, and add a dictionary of **preview_images** with key as **image_1, image_2, image_3, image_4 and image_5**. The values would be the respective paths of the images.

```
render() {
    if (!this.state.fontsLoaded) {
        return <AppLoading />;
    } else {
        let preview_images = {
            "image_1": require("../assets/story_image_1.png"),
            "image_2": require("../assets/story_image_2.png"),
            "image_3": require("../assets/story_image_3.png"),
            "image_4": require("../assets/story_image_4.png"),
            "image_5": require("../assets/story_image_5.png")
        }
```

5. Preview the image selected from the dropdown's value.



6. Add all the text input fields and styling to complete the form. The title field is small and the fields for description, story and morale would be big. Use attributes like multiLine and numberOfLines on the **<TextInput>** component.

    - Use the placeholder attribute for setting the placeholders and specify if the particular textbox is multiline or not. Set the color of the placeholders with a **placeholderTextColor** attribute.

```
<View style={styles.fieldContainer}>
                <TextInput
                  style={styles.inputFont}
                  onChangeText={(title) => this.setState({ title })}
                  placeholder={"Title"}
                  placeholderTextColor="white"
```

```
          />
        </View>
        <View style={styles.fieldContainer}>
          <TextInput
            style={[styles.inputFont, styles.inputFontExtra,
styles.inputTextBig]}
            onChangeText={(description) => this.setState({ description })}
            placeholder={"Description"}
            multiline={true}
            numberOfLines={4}
            placeholderTextColor="white"
          />
        </View>
        <View style={styles.fieldContainer}>
          <TextInput
            style={[styles.inputFont, styles.inputFontExtra,
styles.inputTextBig]}
            onChangeText={(story) => this.setState({ story })}
            placeholder={"Story"}
            multiline={true}
            numberOfLines={20}
            placeholderTextColor="white"
          />
        </View>
        <View style={styles.fieldContainer}>
```

```
                    <TextInput
                      style={[styles.inputFont, styles.inputFontExtra,
styles.inputTextBig]}
                      onChangeText={(moral) => this.setState({ moral })}
                      placeholder={"Moral of the story"}
                      multiline={true}
                      numberOfLines={4}
                      placeholderTextColor="white"
                    />
                  </View>
```

7. Add the relevant styles.

```
inputFont: {
  height: RFValue(40),
  borderColor: "white",
  borderWidth: RFValue(1),
  borderRadius: RFValue(10),
  paddingLeft: RFValue(10),
  color: "white",
  fontFamily: "Bubblegum-Sans"
},
inputFontExtra: {
  marginTop: RFValue(15)
},
inputTextBig: {
  textAlignVertical: "top",
  padding: RFValue(5)
}
```

OUTPUT:

What's next?

In the next class, we'll be working on stack navigation and adding the text-to-speech functionality.

Expand your knowledge:

1. Explore & Experiment with the dropdown component: https://reactnative.dev/docs/picker