Frontend Assignment: Using Animations in React Native App

Sameeksha Verma

**Objective:**

The objective of this assignment is to implement specific animations in a React Native app using native modules in Java.

**Requirements:**

1. **Technology Stack:**

- React Native

- Java (for creating native modules)

- Any additional libraries or tools deemed necessary

**App.js :**

**Purpose**: Serves as the main entry point for the application and sets up navigation using a drawer navigator.

**Content:**

**1. Navigation Setup:** Imports necessary components from React Navigation and sets up a DrawerNavigator using createDrawerNavigator.

**2. Component Rendering:** Renders a NavigationContainer component to wrap the navigation structure.

**3. Drawer Screens:** Defines screen components within the drawer navigator, including:

**- HomeScreen:** Initial screen of the application.

**- FadeIn:** Screen for demonstrating fading in an image with a scaling effect.

**- SlideInList:** Screen for demonstrating slide-in animation for a list of items.

**- RotationButton:** Screen for demonstrating a rotating button animation.

**4. Navigation Container**: Wraps the drawer navigator with NavigationContainer to provide navigation context for the app.

**HomeScreen.js:**

**Purpose:** Represents the home screen of the application.

**Content:**

**Animation Setup:** Initializes an animated value (fadeAnim) to control the opacity of the text.

**Effect Hook:** Triggers a fade-in animation effect when the component mounts, using Animated.timing.

**Rendering:** Renders a view with a background image and an animated text component.

**Background Image:** Displays a background image covering the entire screen.

**Animated Text:** Displays the text "App Animation" with a fade-in effect.

**Styling:** Defines styles for the container, background image, and text components.

**FadeIn.js :**

**Porpose:** The FadeIn component fades in an image with a scaling effect when it is loaded

**Content:**

**1. State Initialization:** In the FadeIn component, we initialize a state variable opacity using the useState hook from React. This opacity variable will be used to control the fading animation.

**2. onLoad Function:** The onLoad function is called when the image is loaded. Inside this function, we define an animation using Animated.timing. This animation gradually increases the opacity of the image from 0 to 1 over a duration of 1500 milliseconds.

**3. Rendering the Component:** The Animated.Image component is used to render the image. We pass the onLoad function to its onLoad prop so that the animation starts when the image is loaded. We also apply animated styles to the image using the opacity value from the state and a scaling effect based on the opacity value.

**4. Navigation Options:** We set the navigation options for the component to specify the header title as 'Fade-In'.

**5. App Component:** The App component renders the FadeIn component wrapped inside an ImageBackground component. This ImageBackground component serves as the background for the FadeIn component. We specify the background image using the source prop and provide styles to ensure the component is centered within the container.

**SlideInList.js :**

**Porpose:** The SlideInList component animates the slide-in effect for a list of items.

**Content:**

**1. Data and Animation Setup:** The component initializes an array of data representing items to be displayed in a list. It also creates an array of animated values (animValues) using useRef, each representing the position of an item on the X-axis. These values start off-screen to the left (-100).

**2. Animation Effect:** The useEffect hook is used to trigger animations when the component mounts. It iterates over the data array and creates animations for each item using Animated.timing. Each animation moves the corresponding animValues[index] from its initial position to 0 (on-screen), with a slight delay for each item.

**3. Rendering Items:** The renderItem function is responsible for rendering each item in the list. It wraps each item in an Animated.View component and applies translation and opacity animations based on the animValues.

**4. Background Image:** The entire component is wrapped in an ImageBackground component, which sets a background image (bgimg2.png). The background image is styled to cover the entire screen (flex: 1) and maintain aspect ratio (resizeMode: 'cover').

**5. Styling:** Styles are defined for the container, content container, and text within the list items. The text is styled with bold font, increased font size, and white color to ensure visibility against the background image.

**6. Navigation Options:** The component is configured with a navigation title of 'Slide-In'.

**RotationButton.js :**

**Purpose:** The RotationButton component animates a button rotation when pressed.

**Content:**

**1. Setup Animation:** The component initializes an animated value (rotationValue) using useRef. This value represents the rotation of the button.

**2. Rotation Function**: The rotateButton function triggers the rotation animation when the button is pressed. It uses Animated.timing to animate the rotationValue from 0 to 1 (one complete rotation), with a duration of 1000 milliseconds and linear easing. After the animation completes, the rotationValue is reset to 0 to prepare for the next rotation.

**3. Interpolation:** The rotateInterpolation variable interpolates the rotationValue to translate it from 0 to 360 degrees. This interpolation is used to dynamically rotate the button.

**4. Animated Styles:** The animatedStyles object applies the rotation transformation to the button using the interpolated value.

**5. Rendering:** The component is rendered within an ImageBackground component, which sets a background image (bgimg3.png). Inside it, a TouchableOpacity wraps the button, applying the rotation animation when pressed.

**6. Styling:** Styles are defined for the button container, button, and button text. The button has a circular shape with a green background, white text, and a border for visual appeal.

**7. Navigation Options:** The component is configured with a navigation title of 'Rotation Button'.