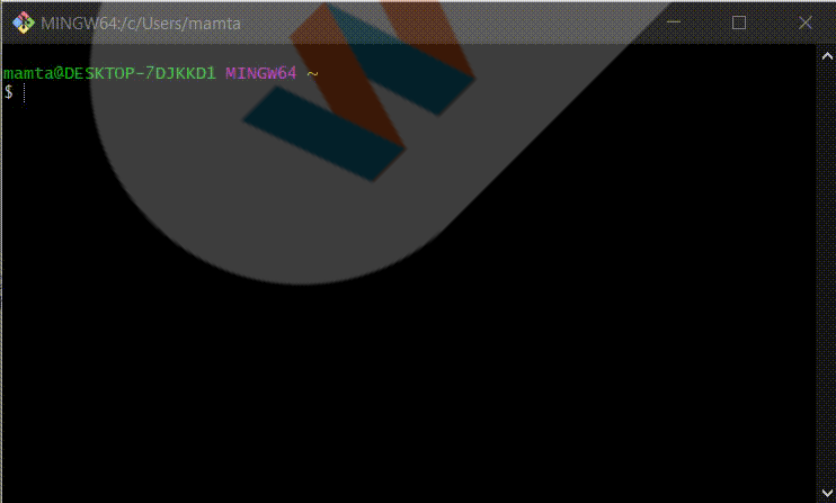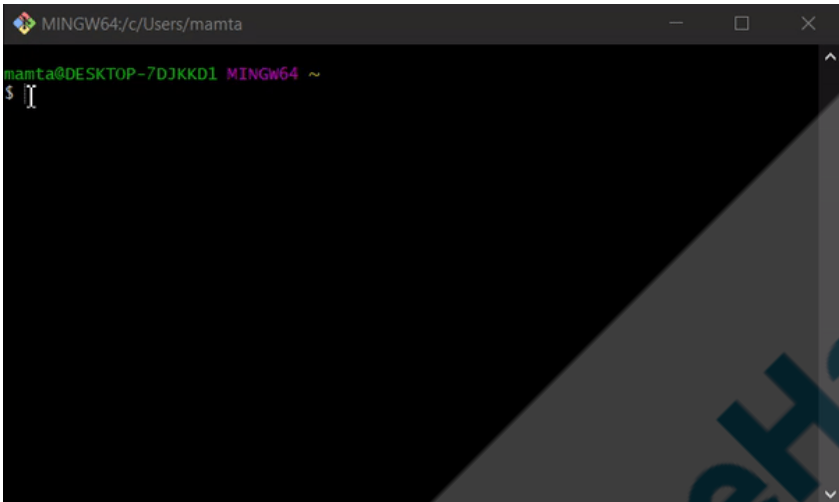| Topic	Monkey-Chunky App - A Case Study
Class Description	Students explore a case study for another app which will help people with reading difficulties practice reading. Students design the UI/UX for the app and build the wireframe for the app including any added game design elements. Students then learn how to take input from the user and display it in the React native environment.
Class	C63
Class time	45 mins
Goal	● Explore case study of an app designed for students with reading difficulties. ● Design wireframe for the app (including UI/UX). ● Collect input from the user and display it on the screen.

Resources Required	● Teacher Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Android/iOS Smartphone with Expo App installed
	
	● Student Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Android/iOS Smartphone with Expo App installed

Class structure	Warm Up	5 mins
	Teacher-led Activity	15 min
	Student-led Activity	15 min
	Wrap up	5 min

WARM-UP SESSION - 5 mins

CONTEXT
● **Explore case study of an app designed for students with reading difficulties.**

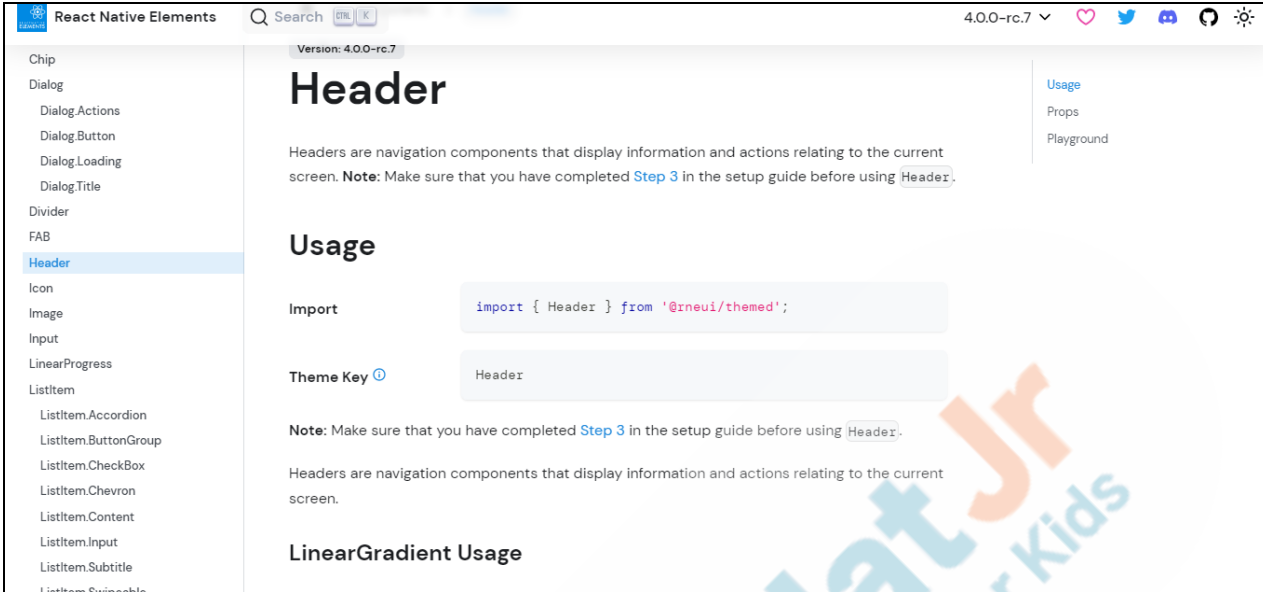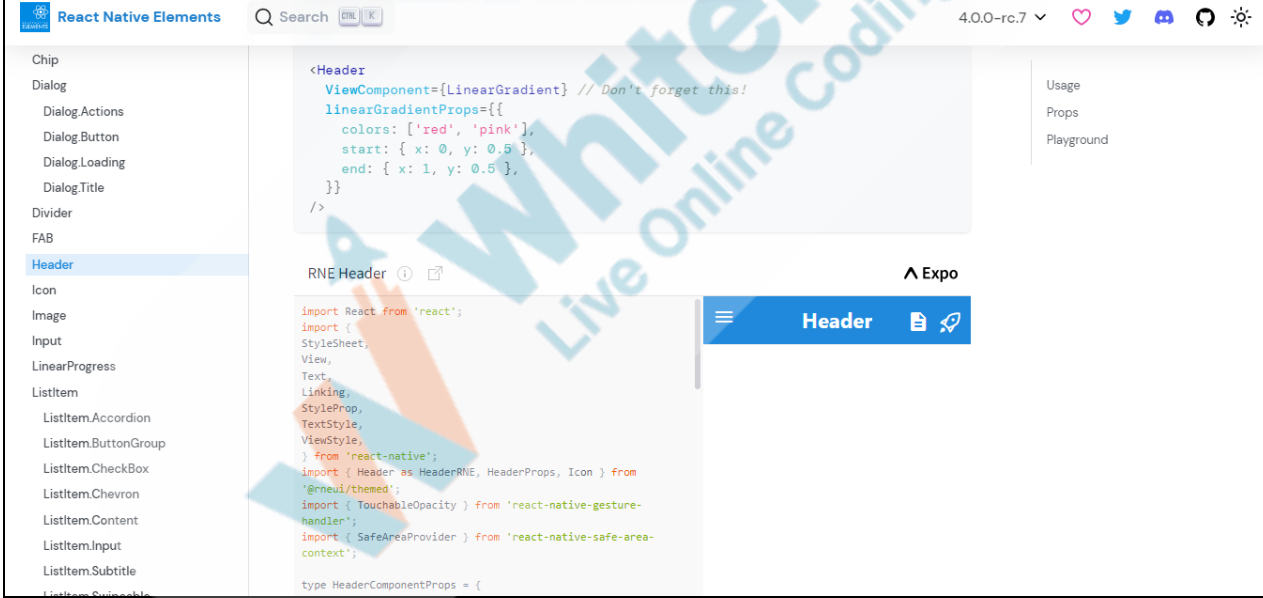|	
Teacher starts slideshow 🖼️ **from slides 1 to 12**	
Refer to speaker notes and follow the instructions on each slide.	

Activity details	Solution/Guidelines
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Run the presentation from slide 1 to slide 4** **Following are the WARM-UP session deliverables:** ● **Greet the student.** ● **Revision of previous class activities.** ● **Quizzes**	**ESR**: Hi, thanks, Yes I am excited about it! Click on the slide show tab and present the slides

QnA Session

Question	Answer
Which command is used to navigate between different folders on the computer? ![MINGW64 terminal window showing: mamta@DESKTOP-7DJKKD1 MINGW64 ~ $] A. mkdir	**B**

B. cd C. npm D. expo	
Which command is used to run your code written in the local environment? MINGW64:/c/Users/mamta mamta@DESKTOP-7DJKKD1 MINGW64 ~ $ A. expo-run B. expo on C. expo npm D. expo start	D

Continue the WARM-UP session	
Activity details	**Solution/Guidelines**
Run the presentation from slide 5 to slide 12 to set the problem statement. **Following are the WARM-UP session deliverables:** ● Appreciate the student. ● Explain Monkey-Chunkey App	Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.

Teacher ends slideshow

	You can watch the video in **Student Activity 2** and read about phonemes in **Student activity 1** to learn more about how word sounds are made up of phonemes.	Student goes through **Student Activity 1 and 2**
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
CHALLENGE ● **Design wireframe / UI / UX for the app.**		
Step 2: Teacher-led Activity (15 min)	What do we do before actually going on to build the app?	ESR: We build a wireframe of how our app will appear.
	Building a wireframe is also called designing the User Interface (UI) or User Experience (UX). It is a crucial part of any app. It defines how your app will be used by the user and how they will experience it. Like coding, designing the user interface/user experience (UI/UX) is also an iterative process. We design a user interface and implement it in the app. After implementation and testing, we might realize that the user experience could be made better by tweaking a few features in our app and we make those changes to the UI/UX. This can go on till we are completely satisfied. Can you spend some time designing	

	the wireframe for our Monkey-Chunky App? You can use a paper and pen to draw the wireframe. Allow the student some time to draw a wireframe for the monkey chunky app.	The student uses a paper and pen to draw a wireframe for the Monkey-Chunky App.
	Can you explain your wireframe and how your user is going to interact with your app?	The student explains the wireframe to the teacher.
	Great! Let's get started on building this app. While we are working on this project, we will learn several new things about React Native and React native Components.	
	We can start our project either on Expo snack online as we were doing for previous projects OR we can do so locally using expo-cli tools we installed in the last class. To start the project locally - you need to type: **expo init <project name>** Choose a blank project and let expo install all the expo libraries for the project. A project directory with your project name will be created. Then change directory (cd) and open the folder in any code editor.	The student chooses whether to code in Expo snack or locally.

	You can also run **expo start** to look at the output of your code. Alternatively, you can also use the online snack expo (**Teacher Activity 3**)	
	Alright, we have seen several in-built React native components so far. We have also built some of our own components like AppHeader.

There are other developers who have also built several react native components and open-sourced them as React Native UI libraries. We can directly import these components in our project and use them. The advantage of using these components is that they are well-designed and thoroughly tested.

One of the popular React Native UI libraries which developers like to use is 'React Native Elements'. You can learn about the different components available and their props, examples on how to use them through the documentation available. | The student goes through the documentation available in **Student Activity 4.** |

	Let's use one of their components, 'Header'. Teacher imports the component from @rneui/themed	The student observes and learns.

	. Note: If student is doing this locally, they have to first use 'npm install react-native-elements' to install the react-native-elements library.	
| ```
1   import * as React from 'react';
2   import { Text, View, StyleSheet } from 'react-native';
3   import { Header } from '@rneui/themed';
4
5   export default class App extends React.Component {
6     render() {
7       return (
8         <View style={styles.container}>
9
10        </View>
11      );
12    }
13  }
14
15  const styles = StyleSheet.create({
16   container: {
17     flex: 1,
18     backgroundColor: '#b8b8b8',
19   }
20  });
21
``` | | |
| | Before using the header component you also need to install another component calls **SafeAreaProvider**

 To install the SafeAreaProvider run the command
npm install
react-native-safe-area-context
if running locally.

 Header component can only be used inside this component only | The student observes how to use Header Component. |

© 2020 - WhiteHat Education Technology Private Limited.

Note: This document is the original copyright of WhiteHat Education Technology Private Limited.

Please don't share, download or copy this file without permission.

| | Let's use the 'Header' component in our App.

Teacher shows how to use the Header Component inside the SafeAreaProvider by referring to the docs and coding in the app. | |

```
1    import * as React from 'react';
2    import { Text, View, StyleSheet } from 'react-native';
3    import { Header } from '@rneui/themed';
4    import { SafeAreaProvider } from 'react-native-safe-area-context';
5
6    export default class App extends React.Component {
7      render() {
8        return (
9
10   <SafeAreaProvider>
11
12       <View style={styles.container}>
13         <Header
14           backgroundColor={'#9c8210'}
15           centerComponent={{
16             text: 'Monkey Chunky',
17             style: { color: '#fff', fontSize: 20 },
18           }}
19         />
20       </View>
21   </SafeAreaProvider>
22       );
23     }
24   }
25
26   const styles = StyleSheet.create({
27     container: {
28       flex: 1,
29       backgroundColor: '#b8b8b8',
30     }
31   });
32
```

My Device | iOS | Android | Web

Monkey Chunky

| | You see, using new React native components is not hard at all!

Now, we need to use a component which will take input from the user. There is a React native component called 'TextInput' for this. Using it is slightly trickier.

Can you take it as a challenge to collect input from the user and display it in your app.?

I will, of course, guide you! | The student takes up the challenge. |

Teacher Stops Screen Share		
STUDENT-LED ACTIVITY - 25 mins		
● **Ask Student to press ESC key to come back to panel** ● **Guide Student to start Screen Share** ● **Teacher gets into Fullscreen**		
ACTIVITY ● **Collect input from the user and display it on the screen.**		
Teacher starts slideshow **from slides 13 to 14** Refer to speaker notes and follow the instructions on each slide.		
	Now it's your turn. Please share your screen with me.	
Teacher ends slideshow		
Step 3: Student-Led Activity (15 min)	We are going to use a new Component called 'TextInput' which is part of React Native library. So let's first import it. *Note:- When the components are they are in lighter shade.* *Once they are used their color changes to a darker shade*	Student imports 'TextInput' from React Native

```
1   import * as React from 'react';
2   import { Text, View, TextInput, StyleSheet } from 'react-native';
3   import { Header } from '@rneui/themed';
4   import { SafeAreaProvider } from 'react-native-safe-area-context';
5
6   export default class App extends React.Component {
7     render() {
8       return (
9
10  <SafeAreaProvider>
11
12        <View style={styles.container}>
13          <Header
14            backgroundColor={'#9c8210'}
15            centerComponent={{
16              text: 'Monkey Chunky',
17              style: { color: '#fff', fontSize: 20 },
18            }}
19          />
20        </View>
21      </SafeAreaProvider>
22    );
23    }
24  }
25
26  const styles = StyleSheet.create({
27   container: {
28      flex: 1,
29      backgroundColor: '#b8b8b8',
30    }
31  });
```

	Let's look at the **TextInput documentation**, the props that are available and how to use them.	Student refers to the **TextInput documentation.**
	TextInput takes the value which is passed on to the value prop. We have to create a state which constantly updates when the user types text as input.	The student assigns some value to the 'TextInput' and sees the output on the screen. The student then creates a state called 'text'. He/She assigns the value of

	The value of 'TextInput' should be the same as 'text state'.	TextInput to the text state. On change of text, the student updates the text state.

```
import { SafeAreaProvider } from 'react-native-safe-area-context';

export default class App extends React.Component {
  constructor() {
    super();
    this.state = {
      text: '',
    };
  }

  render() {
    return (
      <SafeAreaProvider>
        <View style={styles.container}>
          <Header
            backgroundColor={'#9c8210'}
            centerComponent={{
              text: 'Monkey Chunky',
              style: { color: '#fff', fontSize: 20 },
            }}
          />
          <TextInput
            onChangeText={(text) => {
              this.setState({ text: text });
            }}
            value={this.state.text}
          />
        </View>
      </SafeAreaProvider>
    );
  }
}
```

My Device | iOS | Android | **Web**

Monkey Chunky

text

	Let's add some styling to make it look like an Input Text Box.	The student adds styling to text input.

```
          text: 'Monkey Chunky',
          style: { color: '#fff', fontSize: 20 },
        }}
      />
      <TextInput
        style={styles.inputBox}
        onChangeText={(text) => {
          this.setState({ text: text });
        }}
        value={this.state.text}
      />
    </View>
  </SafeAreaProvider>
  );
  }
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#b8b8b8',
  },
  inputBox: {
    marginTop: 200,
    width: '80%',
    alignSelf: 'center',
    height: 40,
    textAlign: 'center',
    borderWidth: 4,
    outline: 'none',
  },
});
```

My Device | iOS | Android | Web

Monkey Chunky

text

	Let's try to render the typed text in 'TextInput' as a normal displayed Text when a button is pressed. Let's create another state called 'displayText' which is to be displayed here.	The student renders a text component which displays a state called 'displayText'.

```jsx
import * as React from 'react';
import { Text, View, TextInput, StyleSheet } from 'react-native';
import { Header } from '@rneui/themed';
import { SafeAreaProvider } from 'react-native-safe-area-context';

export default class App extends React.Component {
  constructor() {
    super();
    this.state = {
      text: '',
      displayText: '',
    };
  }

  render() {
    return (
      <SafeAreaProvider>
        <View style={styles.container}>
          <Header
            backgroundColor={'#9c8210'}
            centerComponent={{
              text: 'Monkey Chunky',
              style: { color: '#fff', fontSize: 20 },
            }}
          />
```

```
render() {
  return (
    <SafeAreaProvider>
      <View style={styles.container}>
        <Header
          backgroundColor={'#9c8210'}
          centerComponent={{
            text: 'Monkey Chunky',
            style: { color: '#fff', fontSize: 20 },
          }}
        />
        <TextInput
          style={styles.inputBox}
          onChangeText={(text) => {
            this.setState({ text: text });
          }}
          value={this.state.text}
        />
        <Text>{this.state.displayText}</Text>
      </View>
    </SafeAreaProvider>
  );
}
}
```

| | Let's create a button called 'Go' button which updates 'displayText' to the same value as text when the user presses this button.

Here TouchableOpacity component will be used to create the button. | The student creates a button which when clicked updates the 'displayText' to 'text state'. |
|---|---|---|

```
                nds ago.  See previous saves.  ✓
16          displayText:  '',
17        };
18      }
19      render() {
20        return (
21          <View style={styles.container}>
22            <Header
23              backgroundColor={'#9c8210'}
24              centerComponent={{
25                text: 'Monkey Chunky',
26                style: { color: '#fff', fontSize: 20 },
27              }}
28            />
29
30            <TextInput
31              style={styles.inputBox}
32              onChangeText={text => {
33                this.setState({ text: text });
34              }}
35              value={this.state.text}
36            />
37            <TouchableOpacity
38              style={styles.goButton}
39              onPress={() => {
40                this.setState({ displayText: this.state.text });
41              }}>
42              <Text style={styles.buttonText}>GO</Text>
43            </TouchableOpacity>
44            <Text style={styles.displayText}>{this.state.displayText}</Text>
45          </View>
46        );
47      }
48    }
```

	Let us add some styling to our buttons and 'displayText'.	The student adds styling to the button and 'displayText'.

```
42    }
43
44    const styles = StyleSheet.create({
45      container: {
46        flex: 1,
47        backgroundColor: '#b8b8b8',
48      },
49      inputBox: {
50        marginTop: 200,
51        width: '80%',
52        alignSelf: 'center',
53        height: 40,
54        textAlign: 'center',
55        borderWidth: 4,
56        outline: 'none',
57      },
58      goButton: {
59        width: '50%',
60        height: 55,
61        alignSelf: 'center',
62        padding: 10,
63        margin: 10,
64      },
65      buttonText:{
66        textAlign: 'center',
67        fontSize: 38,
68        fontWeight: 'bold'
69      },
70      displayText:{
71        textAlign: 'center',
72        fontSize: 38
73      }
74    });
```

iOS Android Web

Monkey Chunky

hero

GO

hero

Teacher Guides Student to Stop Screen Share	
WRAP-UP SESSION - 5 Mins	
Teacher starts slideshow from slide 15 to slide 25	

Activity details	Solution/Guidelines
Run the presentation from slide 15 to slide 25 **Following are the wrap-up session deliverables:** ● **Explain the facts and trivias** ● **Next class challenge** ● **Project for the day** ● **Additional Activity**	Guide the student to develop the project and share with us.

Quiz time - Click on in-class quiz	
Question	**Answer**
Each word in English is made up of a combination of 44 sounds and are known as _____. A. phonemes B. homophones C. heterophones D. phonetics	**A**
Which component helps us to take input from the user? A. input B. InputText C. text D. TextInput	**D**
TextInput component takes the value which is passed on to which of its props?	**B**

A. style
B. value
C. selection
D. placeholder

End the quiz panel

FEEDBACK
- **Encourage the student to explore more of React native documentation and work on the UI of the app.**

Step 4: Wrap-Up (5 min)	Amazing work! We have text input which updates when the user types text. We are also rendering the typed text on the screen. In the next class, we will learn how to break the word into chunks and display it to the user. In the meantime, you can also give some thought to it.	The student listens and thinks about how to chunk the words in the app.
	You can also explore more components available under React native elements. You can add images using Image component in the Monkey-Chunky App to make it more attractive.	The student talks about exploring React Native elements and adding images to the Monkey-Chunky App.
	You get a "hats off". Till next class then. See you. Bye!	Make sure you have given at least 2 Hats Off during the class for: Creatively Solved Activities +10

| **Project Pointers and Cues (5 min)** | *** This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.**

DICTIONARY APP - ONLINE VERSION

Goal of the Project:

In class 63, you learned the use of TextInput instruction to collect text input from a user. You already know how to make API calls to API services in order to get data from them.

You have to use these concepts to create **a simple pocket dictionary app** using which the user can find the definition and meaning of any word.

Story:

Sara and Josh are friends. They are participating in a treasure hunt where the hints are hidden in the meanings of different words. To win this game they definitely need a dictionary to solve the hidden clues.

You have decided to help them by creating a Dictionary App. They can install it on their phones and use it while playing the game. | |

	I am very excited to see your project solution and I know you both will do really well. Bye Bye!	

<table>
<tr><td colspan="3" align="center">Teacher Clicks   ✖ End Class</td></tr>
<tr><td colspan="3" align="center">Teacher ends slideshow</td></tr>
<tr>
<td>Additional Activities</td>
<td>Encourage the student to write reflection notes in their reflection journal using markdown.

Use these as guiding questions:

● What happened today?
  - Describe what happened
  - Code I wrote
● How did I feel after the class?
● What have I learned about programming and developing games?
● What aspects of the class helped me? What did I find difficult?</td>
<td>The student uses the markdown editor to write her/his reflection in a reflection journal.</td>
</tr>
</table>

Activity	Activity Name	Links
Teacher Activity 1	Phonemes description	https://www.dvusd.org/cms/lib/AZ01901092/Centricity/Domain/3795/Sound_Spelling_Chart.pdf

Teacher Activity 2	Phonemes video	https://www.youtube.com/watch?v=wBuA589kfMg
Teacher Activity 3	Class Activity	https://snack.expo.dev/@procodingclass/monkey-chunky-stage-1
Teacher Activity 4	React native elements library	https://reactnativeelements.com/docs
Teacher Activity 5	Text Input Documentation	https://facebook.github.io/react-native/docs/textinput
Teacher Activity 6	Reference link (Use Latest Snack SDK Version)	https://snack.expo.dev/@procodingclass/monkey-chunky-stage-1:-reference
Student Activity 1	Phonemes description	https://www.dvusd.org/cms/lib/AZ01901092/Centricity/Domain/3795/Sound_Spelling_Chart.pdf
Student Activity 2	Phonemes video	https://www.youtube.com/watch?v=wBuA589kfMg
Student Activity 3	Class Activity (Use Latest Snack SDK Version)	https://snack.expo.dev/@procodingclass/sa3-monkey-chunky-stage-1
Student Activity 4	React native elements library	https://reactnativeelements.com/docs
Student Activity 5	Text Input Documentation	https://facebook.github.io/react-native/docs/textinput
Project Solution	Dictionary App-Online Version	https://github.com/priyapandey2020/aaab251f19dfa78857b6b445445275

		87
Teacher Reference visual aid link	Visual aid link	https://s3-whjr-curriculum-uploads.whjr.online/53bbbc84-e056-4c3b-87df-0592874f5ab7.html
Teacher Reference In-class quiz	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/8c25669a-2336-44b6-9fdb-697bf256d08b.pdf