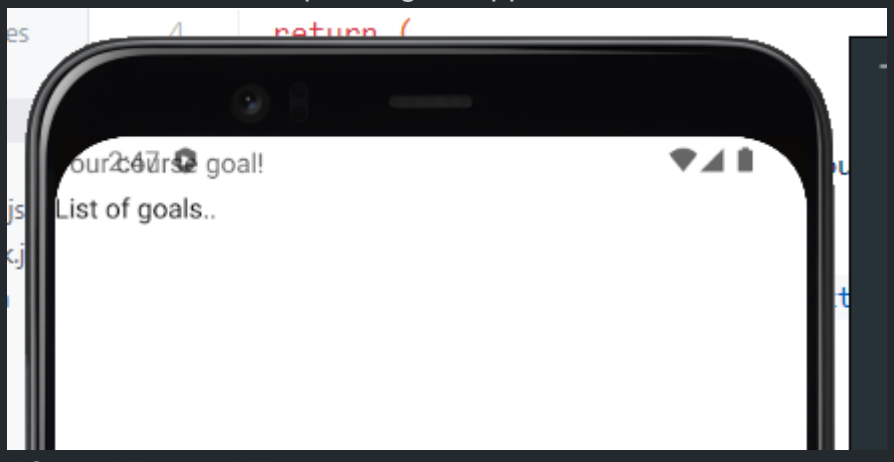
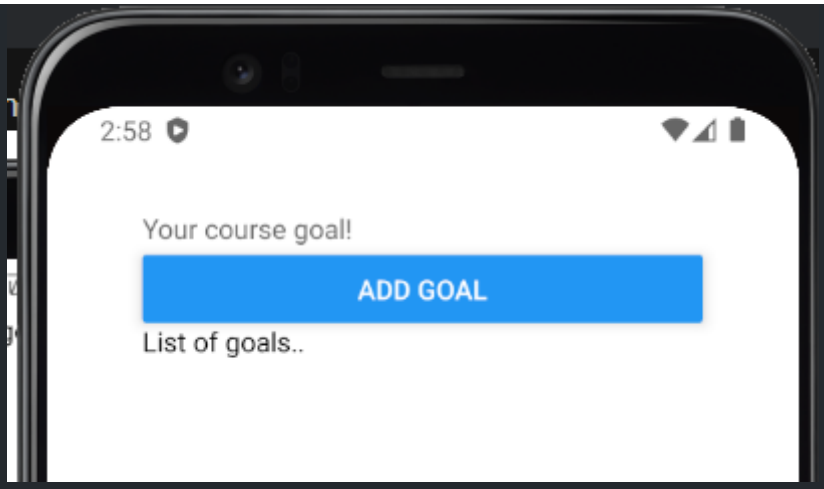
Layouts and Flexbox

```
import { StyleSheet, Text, View, Button, TextInput} from "react-native";

export default function App() {
  return (
    // methana styles.appcontainer eke nethu unoth wena eke pahala preview eke tiyei
    <View style={styles.appCointainer}>
      <View>
        {/* before use any component like TextInput we should import from "react-native".
         It is done by very first line*/}
        <TextInput placeholder="Your course goal!"/>
        <Button title="Add Goal"/>
      </View>
      <View>
        <Text>List of goals..</Text>
      </View>
    </View>
  );
}

const styles = StyleSheet.create({
  appCointainer:{
    padding:50
  }

});
```

- If there isn't define a padding for appCointainer view is like this

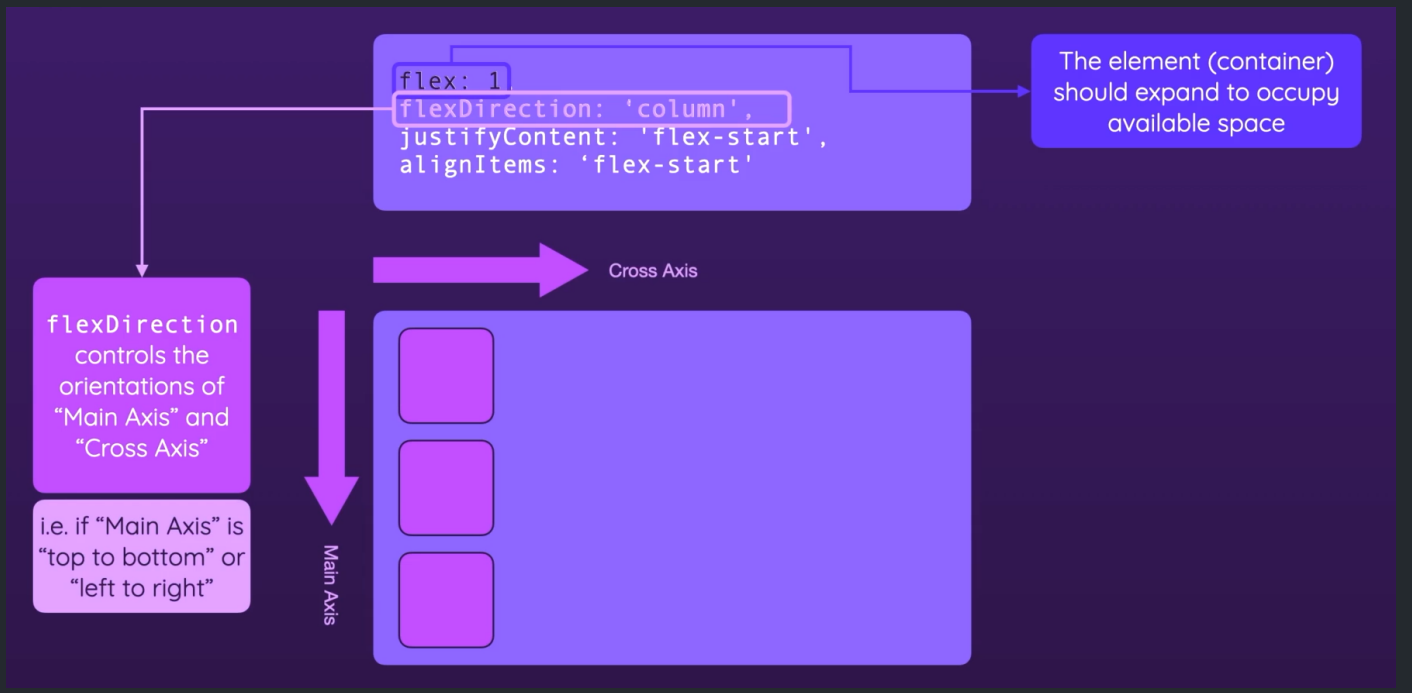


- After adding padding



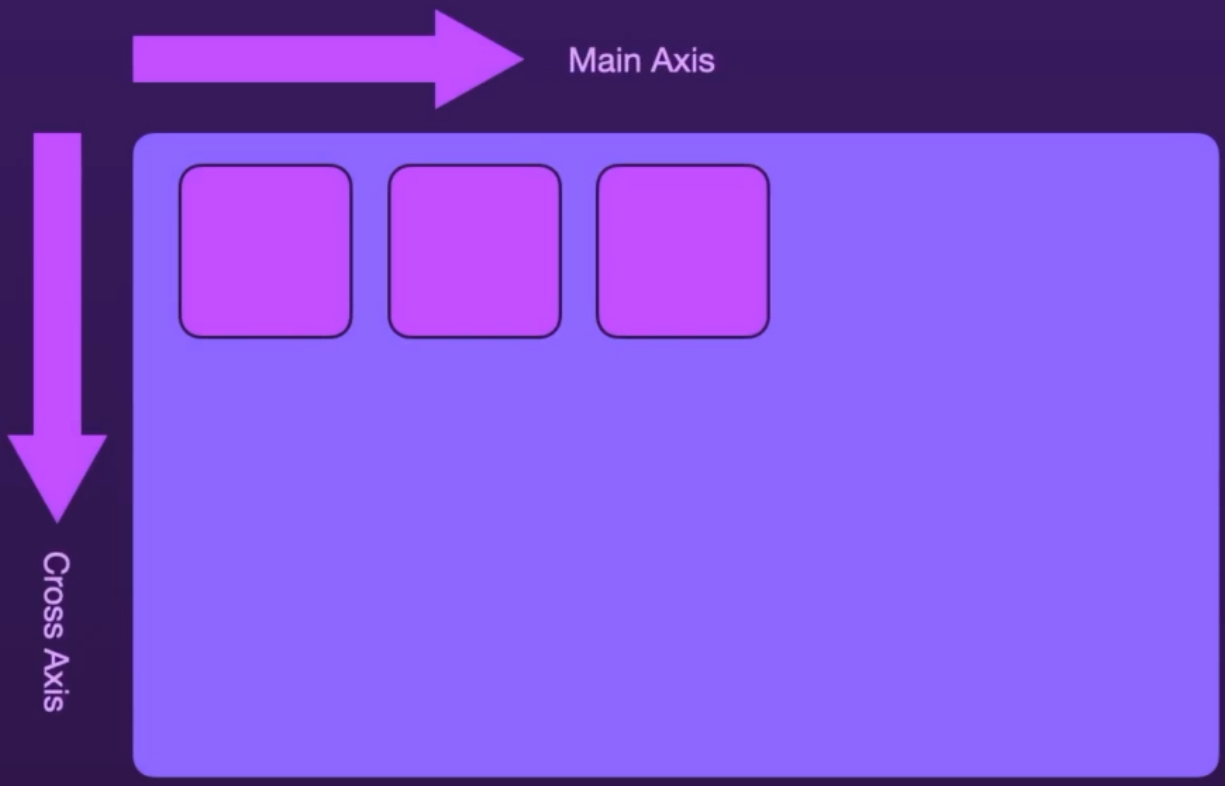
Flexbox

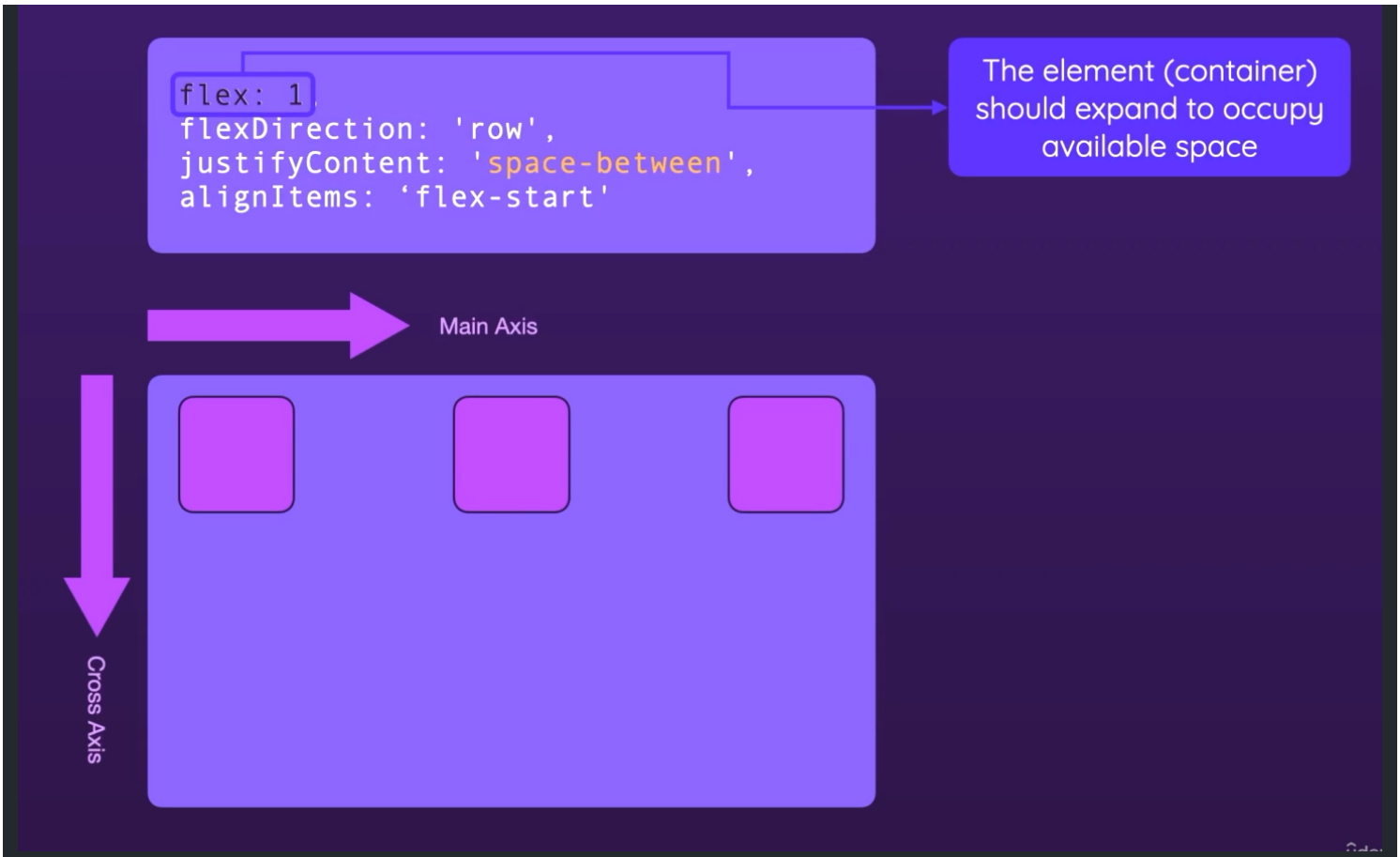
- positioning elements inside a container



- Not like CSS here default flex direction is Colmn

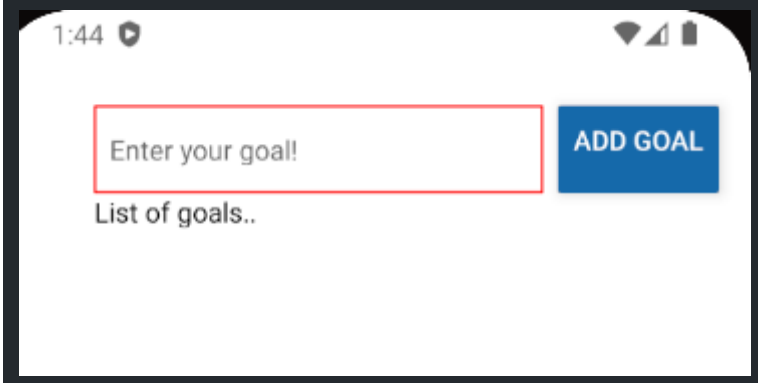
```
flex: 1,
flexDirection: 'row',
justifyContent: 'flex-start',
alignItems: ‘flex-start'
```





- Applying after justifyContent: 'space-between'

```jsx
export default function App() {
  return (
    <View style={styles.appCointainer}>
      {/* We are going to apply flexbox to this View to controll
      the position of TextInput and Button */}
      <View style={styles.flexView}>
        <TextInput style={styles.textInput} placeholder="Enter your goal"/>
        <Button title="Add Goal"/>
      </View>
      <View>
        <Text>List of goals..</Text>
      </View>
    </View>
  );
}

const styles = StyleSheet.create({
  appCointainer:{
    padding:50
  },
  flexView:{
    flexDirection: 'row',
    justifyContent: 'space-between'
  },
  textInput:{
    borderWidth: 1,
    borderColor:'red',
    // take 80% width from available space **wrap those numbers within ''
    width:'80%',
    marginRight: 8,
    padding: 8
  }

});
```



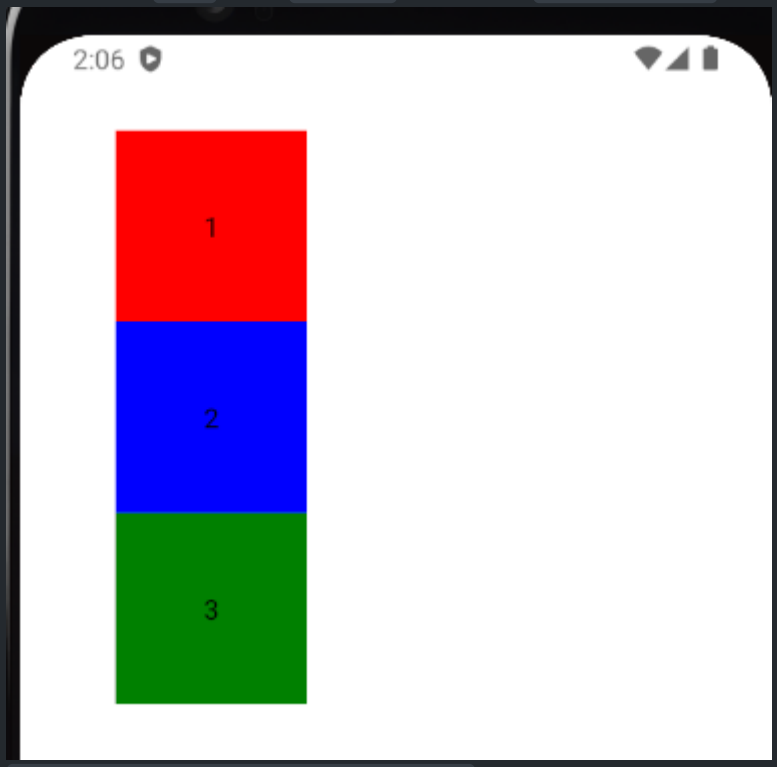
Dive deeper into flexbox

- Documentation of flexbox- https://reactnative.dev/docs/flexbox
- Every `View` by default organize it's children using flexbox
- Here default flex direction is **column**.

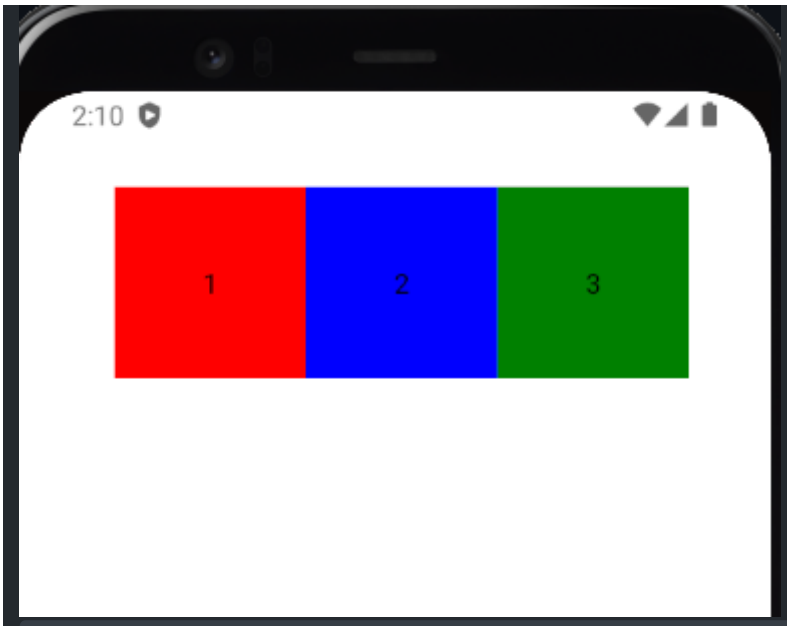
```
// Change default flex direction from column to row
    <View style={{ padding: 50, flexDirection:'row' }}>
```

`//Here this style is apply for the parent View`

- excepting `row` and `column` we have `row-reverse` and `column-reverse`



`<View style={{ padding: 50}}>`

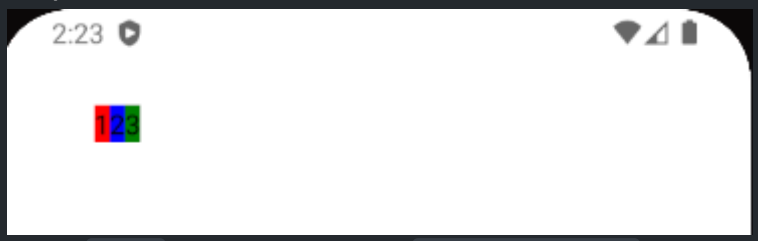



```
<View style={{ padding: 50, flexDirection:'row' }}>
```

- Above mention screen shots's code contain 100px width and height in every child `View`. we will remove all those sizes and see the output

```
  export default function App() {
  return (
    <View style={{ padding: 50, flexDirection:'row' }}>
      <View
        style={{
          backgroundColor: "red",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
        <Text>1</Text>
      </View>
      <View
        style={{
          backgroundColor: "blue",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
        <Text>2</Text>
      </View>
      <View
        style={{
          backgroundColor: "green",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
        <Text>3</Text>
      </View>
    </View>
  );
}
```

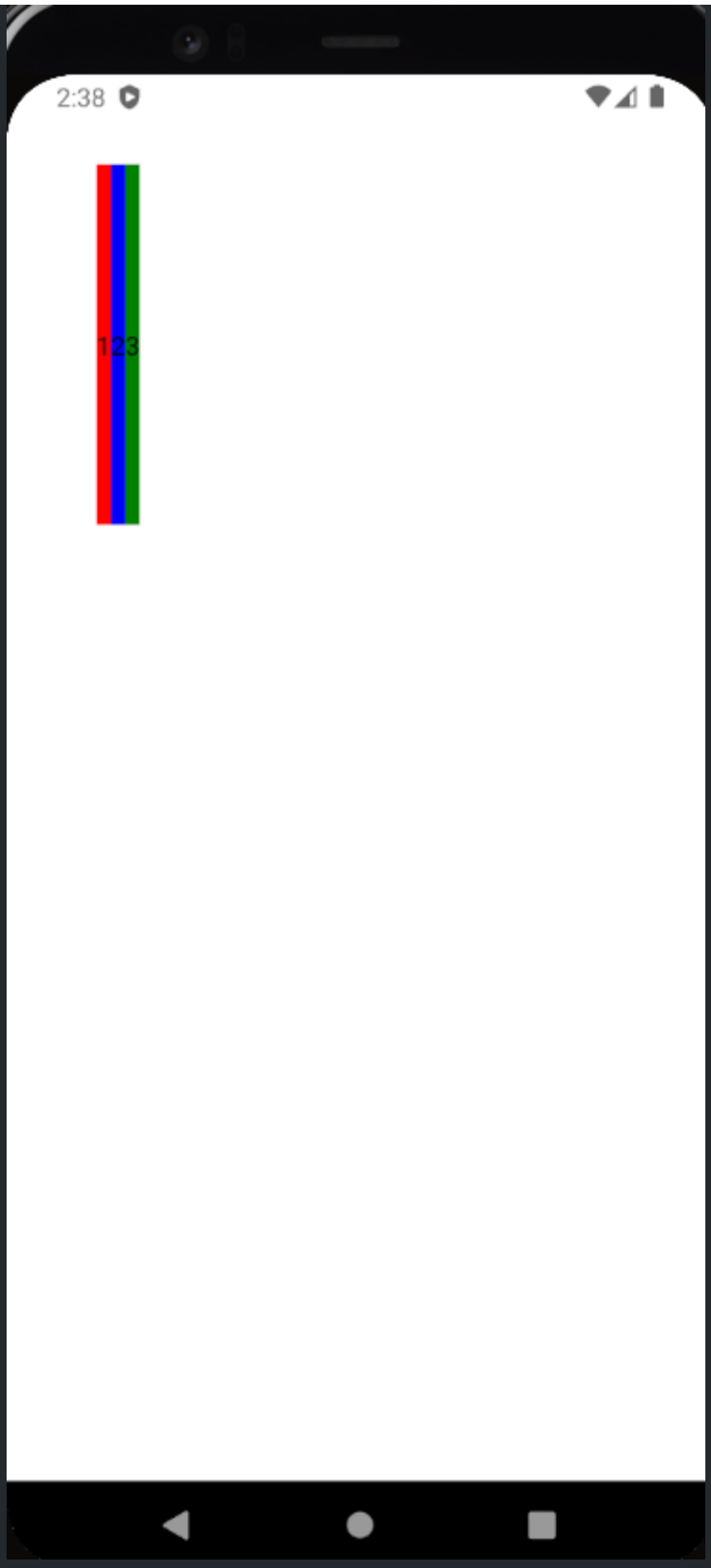
- outputs will be look like



- Here `view` only took size of `<Text>1</Text>` . If there is no **text** then there is no color view
- Then we will add height and width for only parent `View`

```
<View style={{ padding: 50, flexDirection:'row', width:'80%',height:300}}>
```

- output is



- Here there is no impact with styles
- When working with flexbox there is 2 axis(**Main axsis and Cross axsis**). Main axsis is depends on flex direction

  1.If flex direction is *row* then main axsis is **from left to right**.

  2.If flex direction is *column* then main axsis is **top to bottom**.

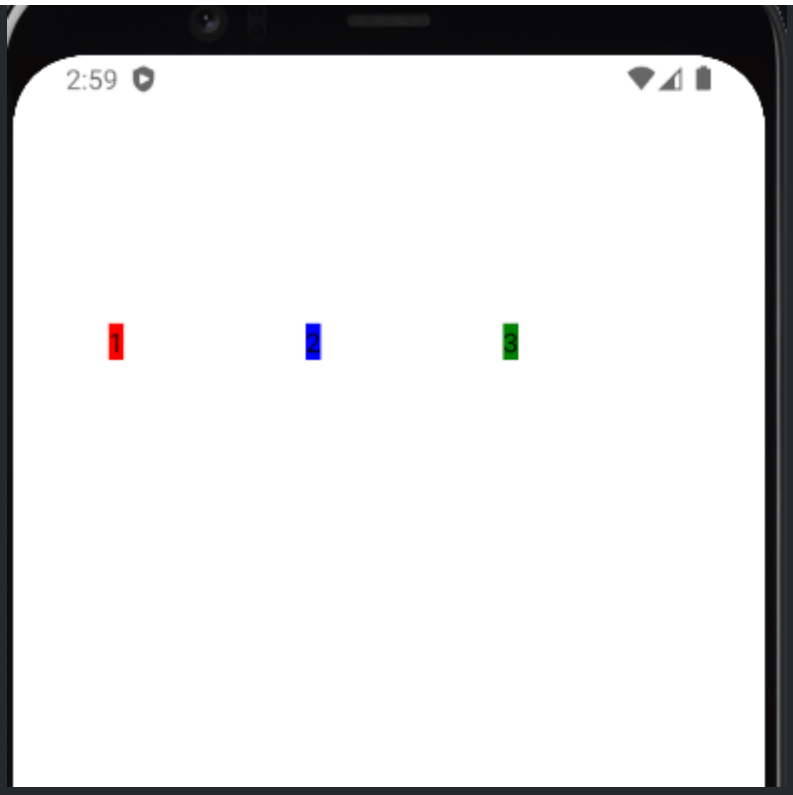
- *For reverse properties direction will be inverted.*
  It means if the flexdirection is `row-reverse` *then main axsis will be* **rigtht to left**.

```
//This is also same parent View
   <View
     style={{
       padding: 50,
       flexDirection: "row",
       width: "80%",
       height: 300,
       justifyContent: "space-between",
       alignItems: "center",
     }}
   >
```

- For organize element throw the *main axis* we can use `justifyContent:` . And throw the *Cross axis* we can use `alignItems:`

```
//This is also same parent View
    <View
      style={{
        padding: 50,
        flexDirection: "row",
        width: "80%",
        height: 300,
        justifyContent: "space-between",
        alignItems: "center",
      }}
    >
```

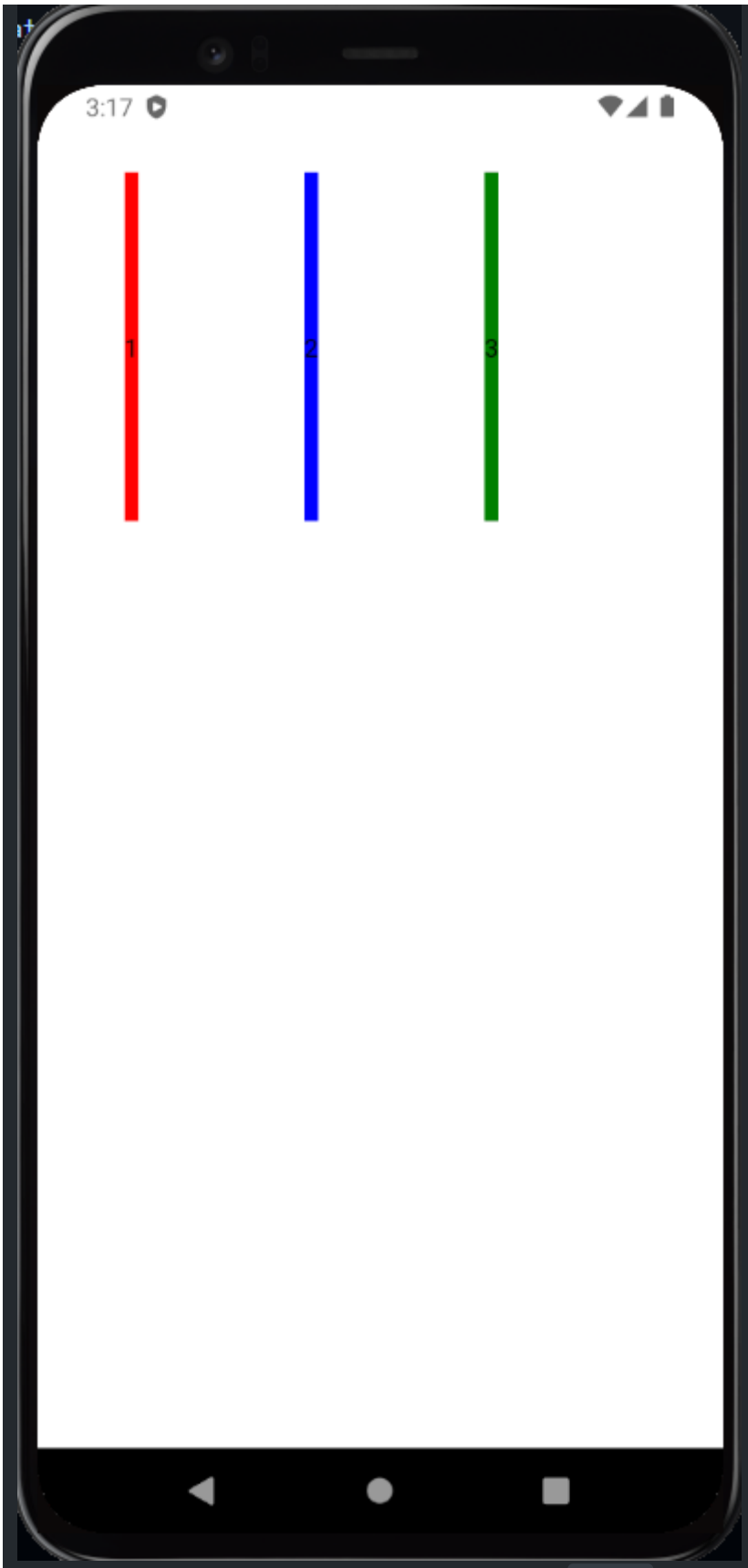
- output is



Here we can see having space between item along the **main axis** and item became center along the **cross axsis**. But not like previous code (before applying `justifyContent` and `alignItem`) height of the `View` is reduce.
Reson for that is at default `alignItem` has `'stretch'

- After applying `'stretch'`

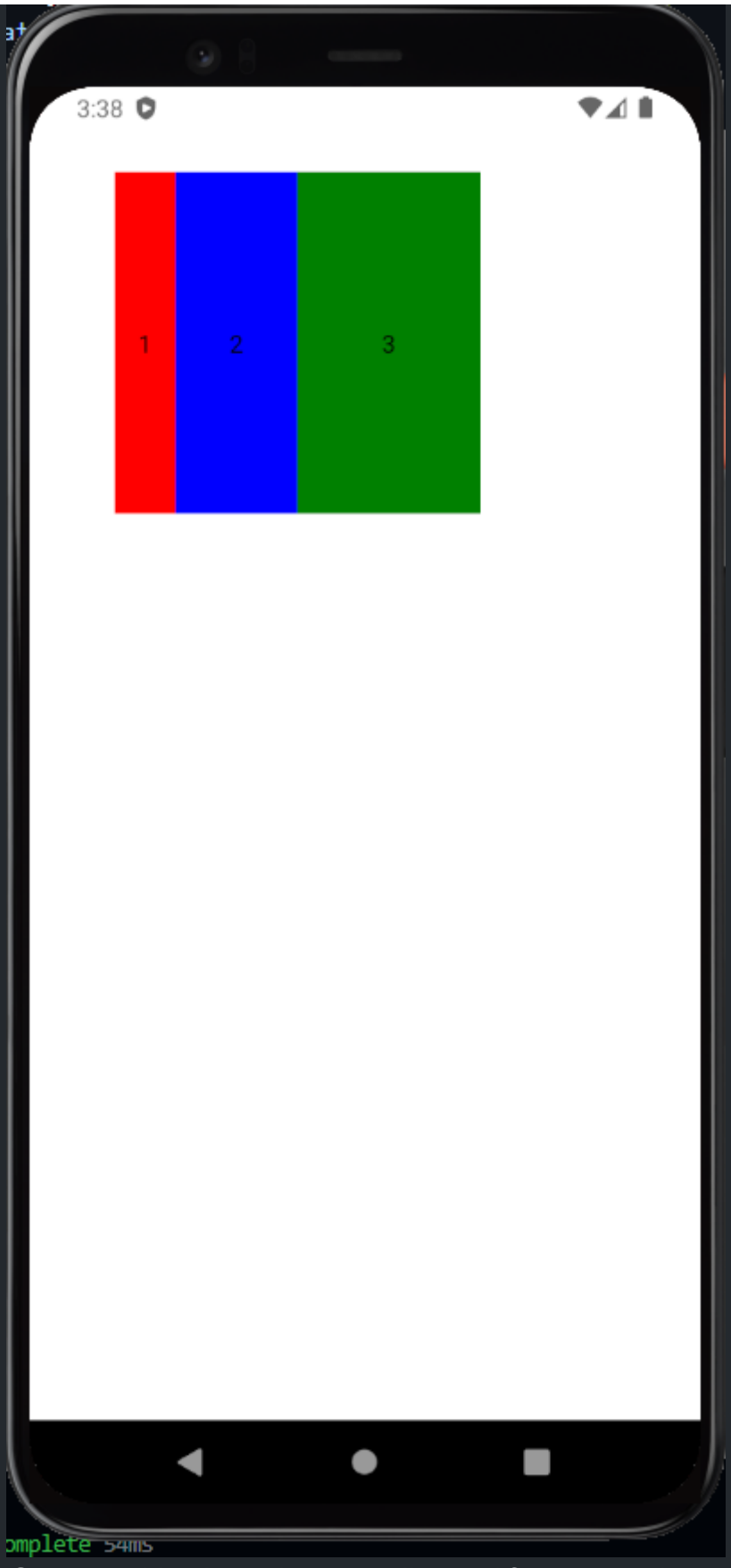
```
<View
      style={{
        padding: 50,
        flexDirection: "row",
        width: "80%",
        height: 300,
        justifyContent: "space-between",
        alignItems: 'stretch',
      }}
    >
```



- Available width ekema tiyena vidiyata `views`
  3 na ganna we can't use `'stretch'`. Because it isn't property of *justifyContent*
- To configure that we need to apply a style in `child Event`.

```
<View
      style={{
        padding: 50,
        flexDirection: "row",
        width: "80%",
        height: 300,
        justifyContent: "space-between",
        alignItems: 'stretch',
      }}
    >
      <View
        style={{
          flex:1, //flex is using for adjust width
          backgroundColor: "red",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
        <Text>1</Text>
      </View>
      <View
        style={{
          flex:2,
          backgroundColor: "blue",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
        <Text>2</Text>
      </View>
      <View
        style={{
          flex:3,
          backgroundColor: "green",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
```

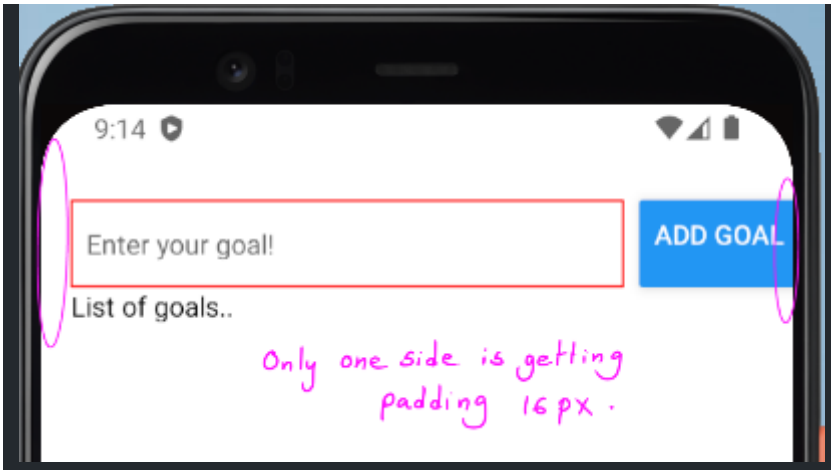
- output is



Get some idea how sizes are allocated from example

```
import { StyleSheet, Text, View, Button, TextInput} from "react-native";

export default function App() {
  return (
    <View style={styles.appCointainer}>
      <View style={styles.flexView}>
        <TextInput style={styles.textInput} placeholder="Enter your goal!"/>
        <Button title="Add Goal"/>
      </View>
      <View>
        <Text>List of goals..</Text>
      </View>
    </View>
  );
}

const styles = StyleSheet.create({
  appCointainer:{
    paddingTop:50,
    paddingHorizontal:16,//apply padding horizontally
  },
  flexView:{
    flexDirection: 'row',
    justifyContent: 'space-between'
  },
  textInput:{
    borderWidth: 1,
    borderColor:'red',
    // take 80% width from available space
    width:'80%',
    marginRight: 8,
    padding: 8
  }

});
```

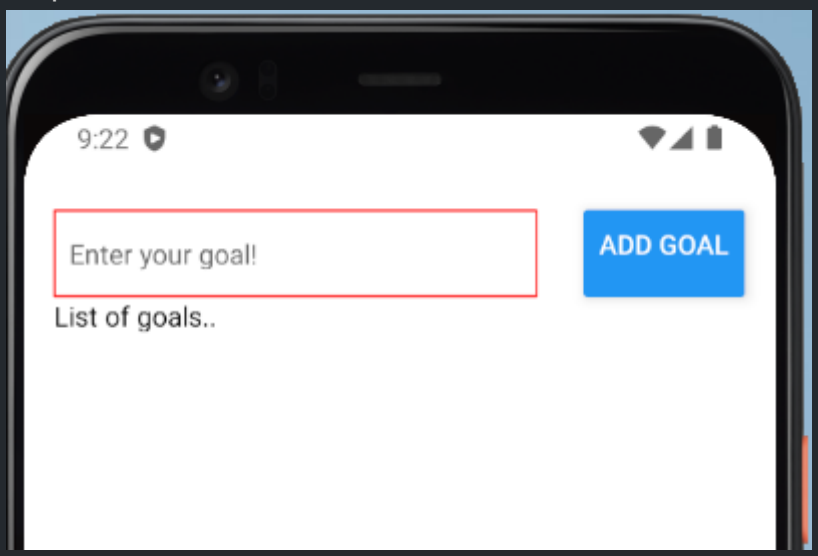
Here we apply `padding` for main `View`. But it'sn't affected to one side



Reason for this is default size of button. There for we are going to decrees **width** of `TextInput`

```
textInput:{
    borderWidth: 1,
    borderColor:'red',
    // take 70% width from available space
    width:'70%',
    marginRight: 8,
    padding: 8
  }

  //preview eke wetila tiyana eke don't care "<p data-line="301" class="sync-line" style="margin:0;"></p>"
```

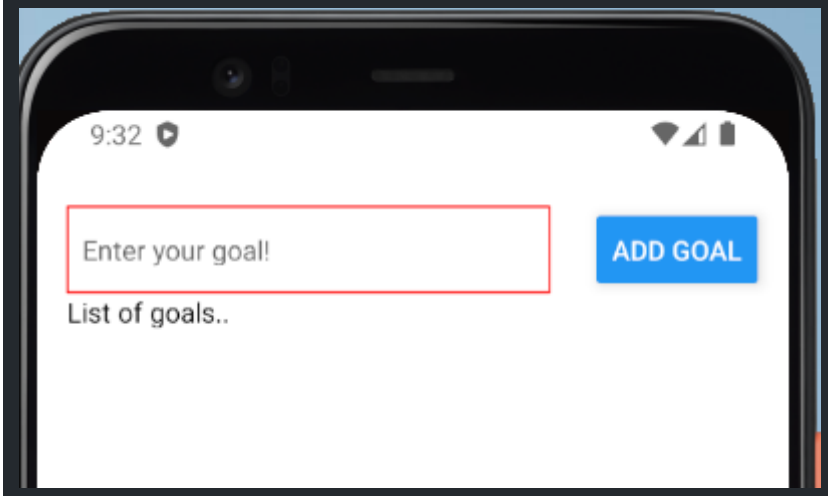
- output is



- But here we can see text in the button is not in center. Here button is `Stretch` according to the height of **TextInput**. To avoid those default styles

```
flexView:{
    flexDirection: 'row',
    justifyContent: 'space-between',
    alignItems:'center',//to align text in the button center
  },
```

Here is new output



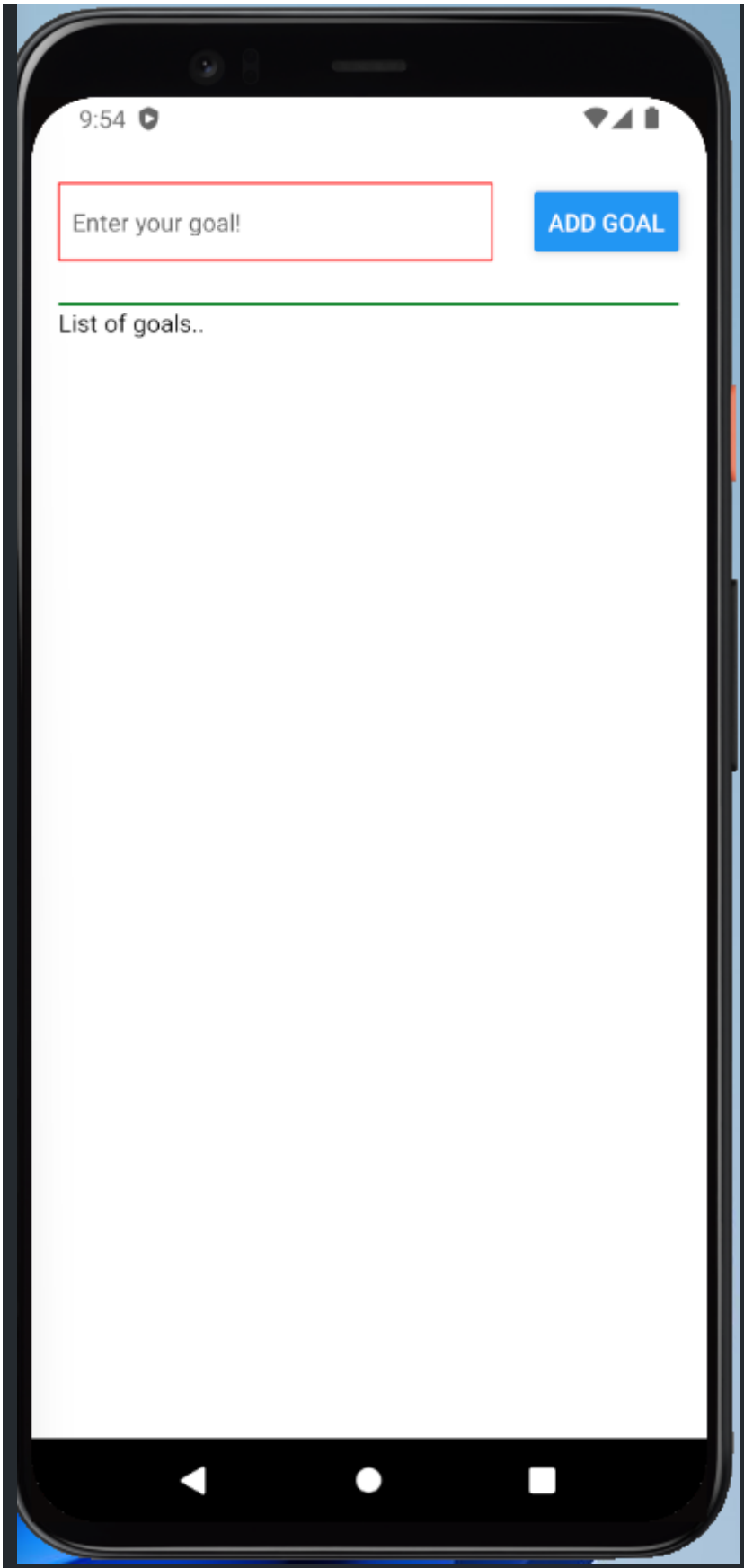
Now the button is ok

Here we can't apply styles to `button` for center the text. Because button is a component which *doesn't have* `style props`

https://reactnative.dev/docs/text Component documentation eke relevent component page eke **Right side** tiyenawa support karana props

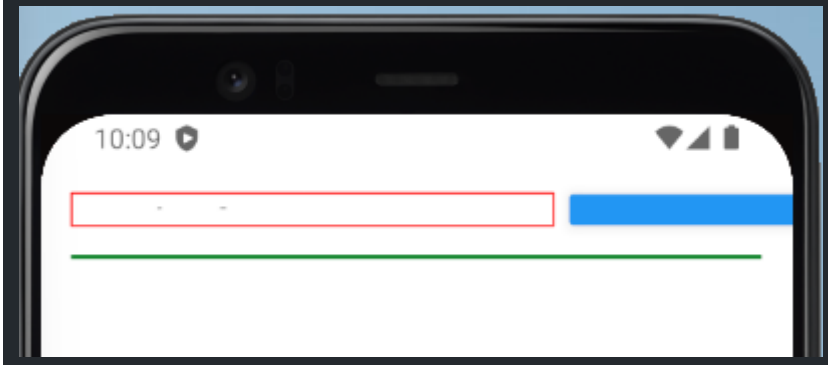
- Now we want to add some gap and **line** between *TextInput* and *List of goals..*

```
flexView:{
    flexDirection: 'row',
    justifyContent: 'space-between',
    alignItems:'center',
    paddingBottom: 24,// adding a gap
    borderBottomWidth: 2, //only adding border for bottom get a horizontal line
    borderBottomColor: '#198a35',
  },
```



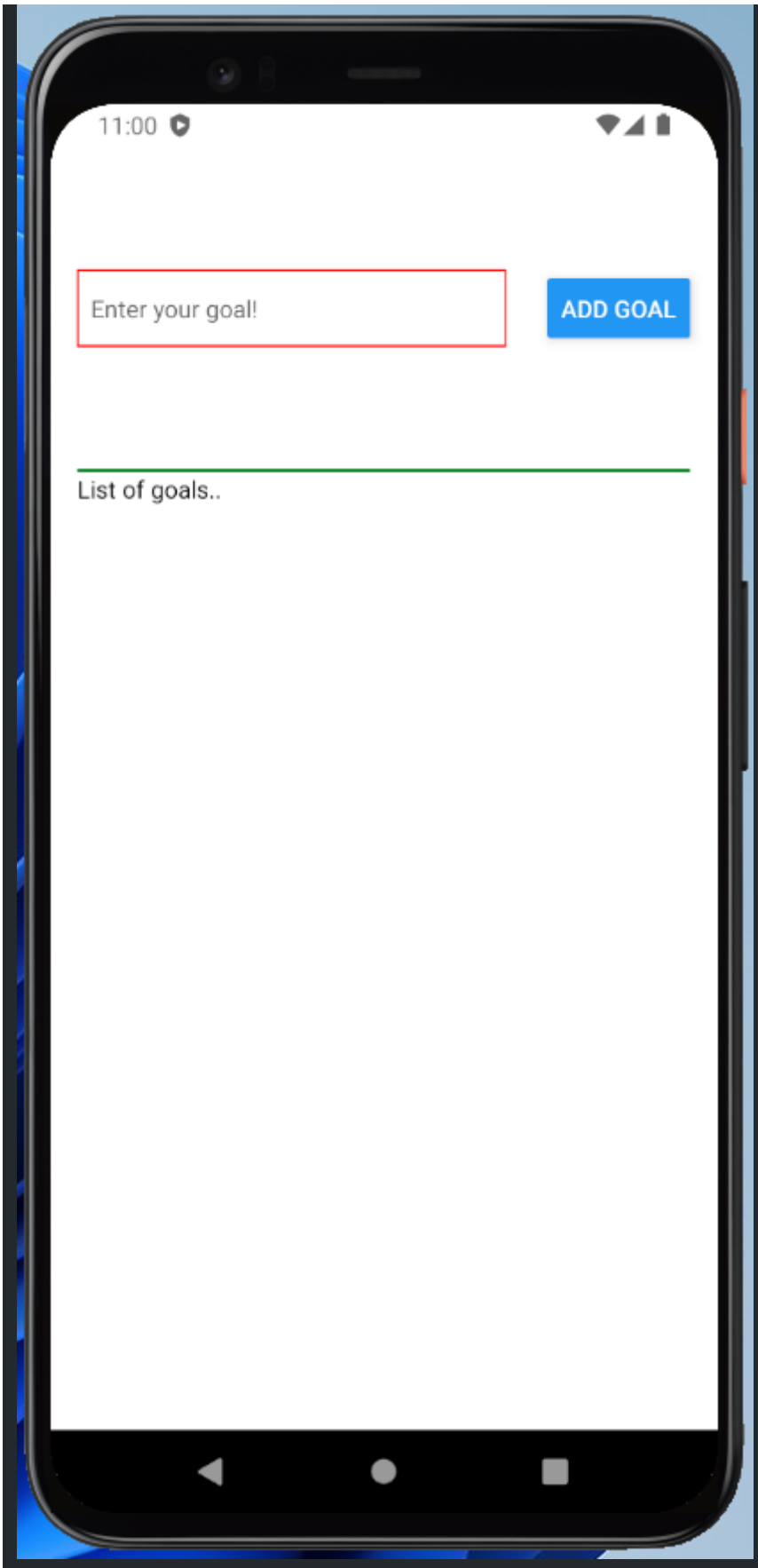
- Now we need to get height of `TextInputArea` as 1/5 in screen available height. To get that we put `flex:1` for `flexView` View and `flex:5` for list of goals...'s View. **But it will mess the work**.

```
export default function App() {
  return (
    <View style={styles.appCointainer}>
      <View style={styles.flexView}>
        <TextInput style={styles.textInput} placeholder="Enter your goal!"/>
        <Button title="Add Goal"/>
      </View>
      <View style={styles.goalContainer}>//applying flex:4 in styling
        <Text>List of goals..</Text>
      </View>
    </View>
  );
}

const styles = StyleSheet.create({
  appCointainer:{
    paddingTop:50,
    paddingHorizontal:16,
  },
  flexView:{
    flex: 1, // to achive 1/5
    flexDirection: 'row',
    justifyContent: 'space-between',
    alignItems:'center',
    paddingBottom: 24,
    borderBottomWidth: 2,
    borderBottomColor: '#198a35',
  },
  textInput:{
    borderWidth: 1,
    borderColor:'red',
    width:'70%',
    marginRight: 8,
    padding: 8,
  },
  goalContainer:{
    flex:4
  }
```



- Reson for beeing like this outer Container `styles.appCointainer` needs to take **Entire available screen height**. But in default this container only took as much height it needs. To get all available height we just put `flex:1` for `app.container`

```
export default function App() {
  return (
    <View style={styles.appCointainer}>
      <View style={styles.flexView}>
        <TextInput style={styles.textInput} placeholder="Enter your goal!"/>
        <Button title="Add Goal"/>
      </View>
      <View style={styles.goalContainer}>
        <Text>List of goals..</Text>
      </View>
    </View>
  );
}

const styles = StyleSheet.create({
  appCointainer:{
    flex:1, //now take whole available screen size
    paddingTop:50,
    paddingHorizontal:16,
  },
  flexView:{
    flex: 1,
    flexDirection: 'row',
    justifyContent: 'space-between',
    alignItems:'center',
    paddingBottom: 24,
    borderBottomWidth: 2,
    borderBottomColor: '#198a35',
  },
  textInput:{
    borderWidth: 1,
    borderColor:'red',
    width:'70%',
    marginRight: 8,
    padding: 8,
  },
  goalContainer:{
    flex:4
  }


});
```


11:00
Enter your goal!
ADD GOAL
List of goals..