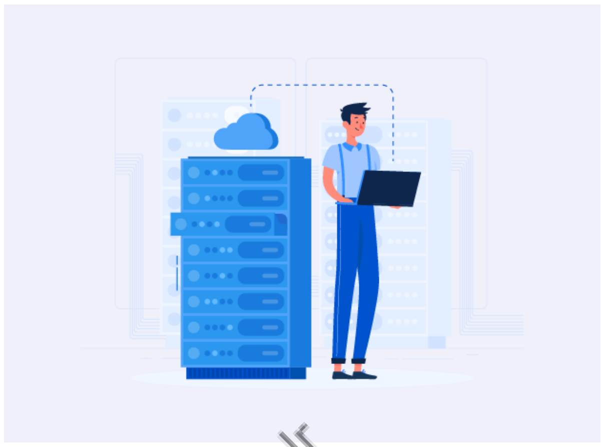
Exoplanet Catalog



What is our GOAL for this MODULE?

The goal of this module is to create a react native App, which will act as a catalog to display data that we have in our Flask server.

What did we ACHIEVE in the class TODAY?

- Created screen 1 (To display list of all the planets)
- Created screen 2 (To display list)
- Tested the API

Which CONCEPTS/CODING BLOCKS did we cover today?

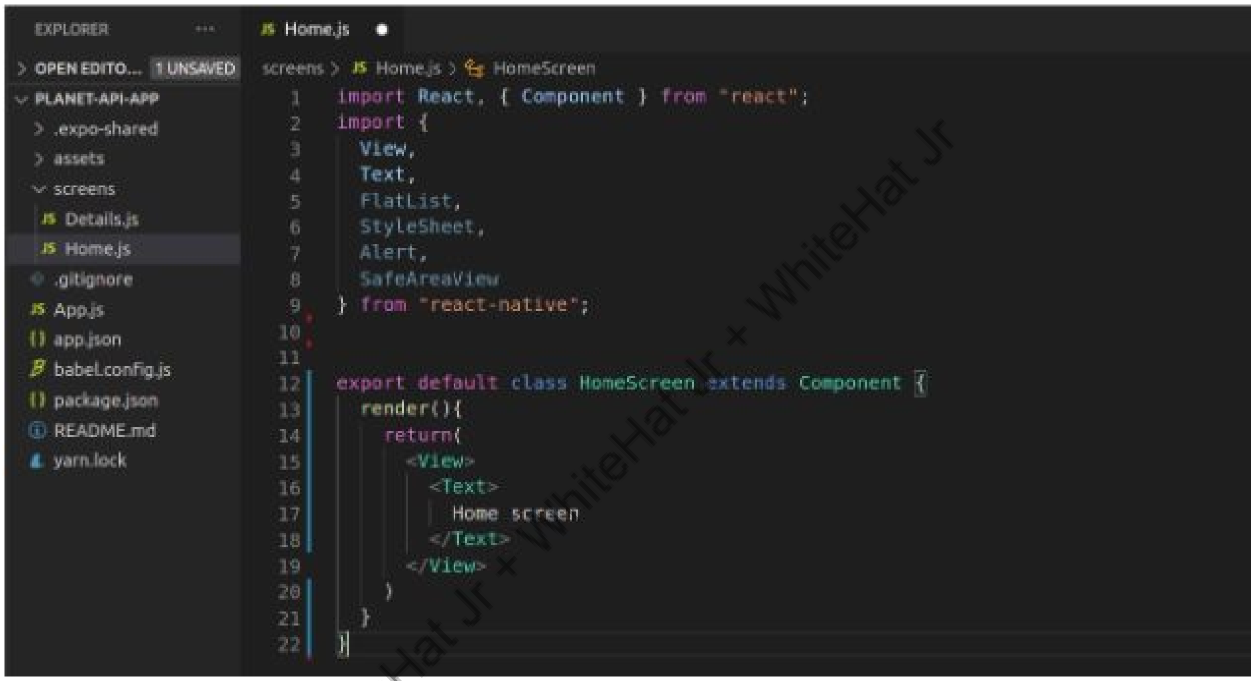
- Flask
- React Native
- API Integration

How did we DO the activities?

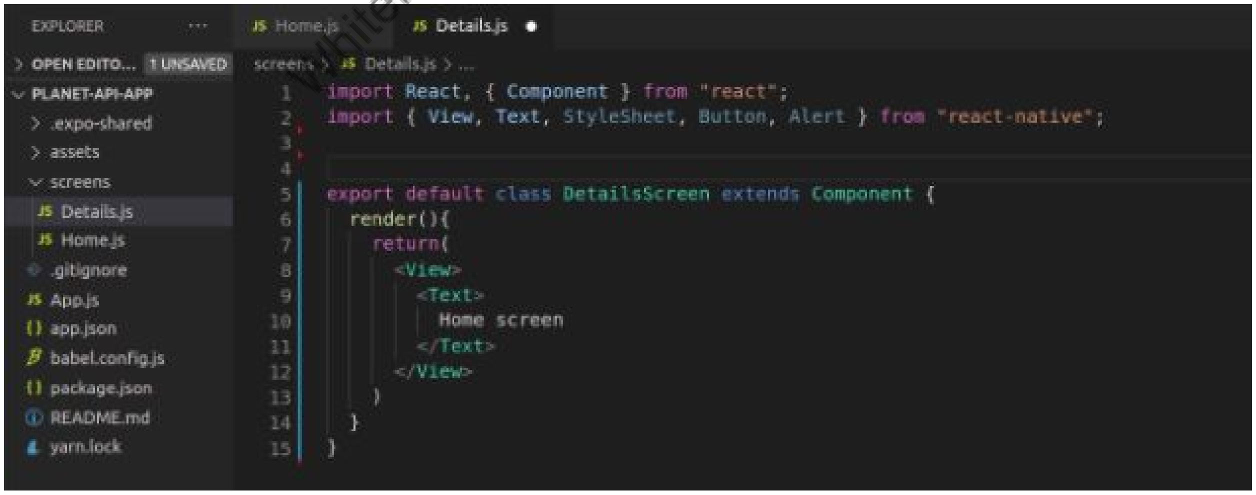
1. Initiate a React Native App with Expo

   expo init Planet-App

2. Open the code in VS Editor.
3. Create 2 blank screens **Home.js & Details.js.** Here, Home.js will contain the list of all the planets and Details.js will contain the list of selected planets.



```
import React, { Component } from "react";
import {
  View,
  Text,
  FlatList,
  StyleSheet,
  Alert,
  SafeAreaView
} from "react-native";


export default class HomeScreen extends Component {
  render(){
    return(
      <View>
        <Text>
          Home screen
        </Text>
      </View>
    )
  }
}
```

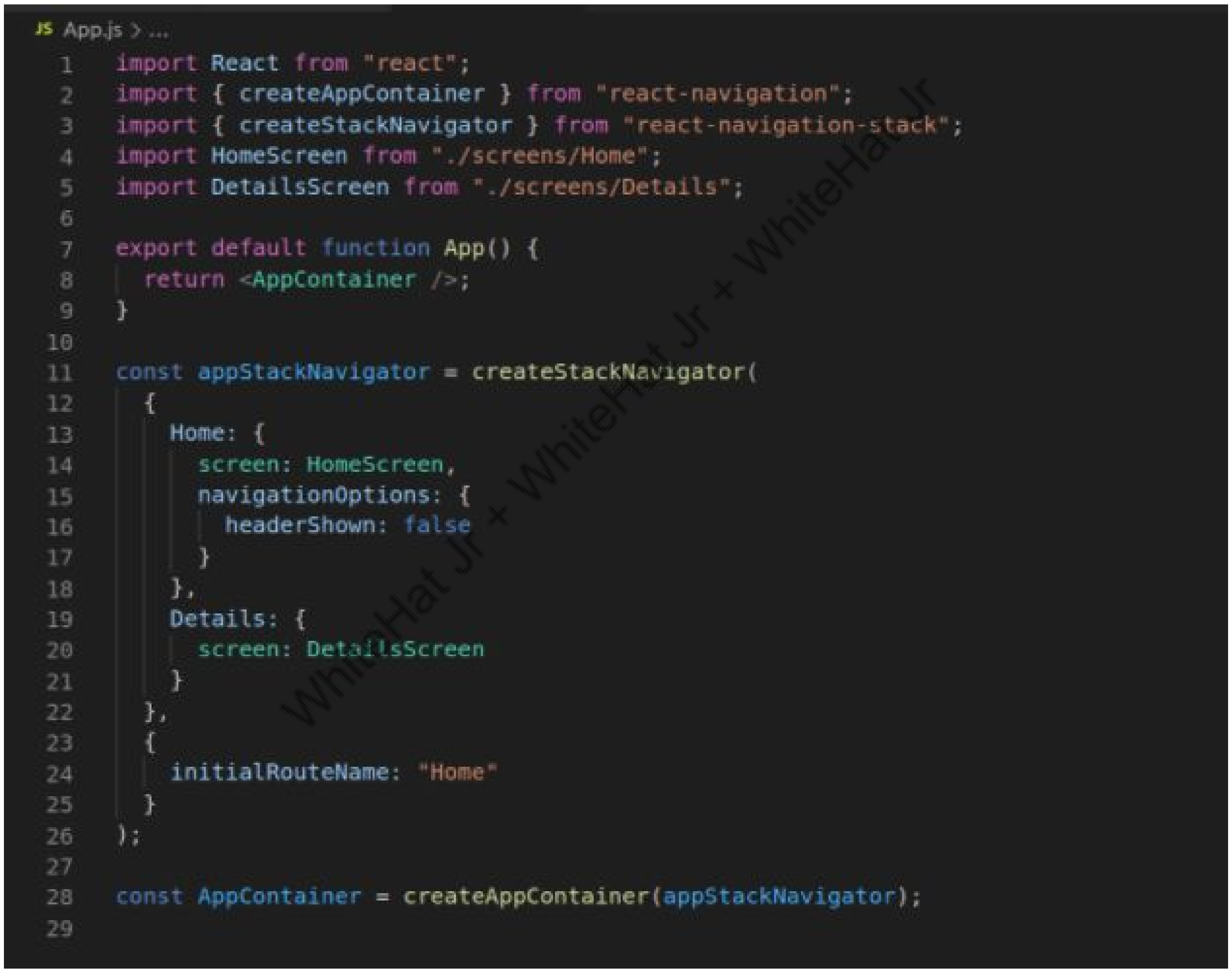


```
import React, { Component } from "react";
import { View, Text, StyleSheet, Button, Alert } from "react-native";


export default class DetailsScreen extends Component {
  render(){
    return(
      <View>
        <Text>
          Home screen
        </Text>
      </View>
    )
  }
}
```

4. Install dependencies for react-navigation.

   npm install react-navigation
   npm install react-navigation-stack

5. Import ***createAppContainer*** from ***react-navigation*** and ***createStackNavigator*** from ***react-navigation-stack.***
6. Import Home.js and Details.js from the screen folder.
7. Create the StackNavigator and add the screens to it.

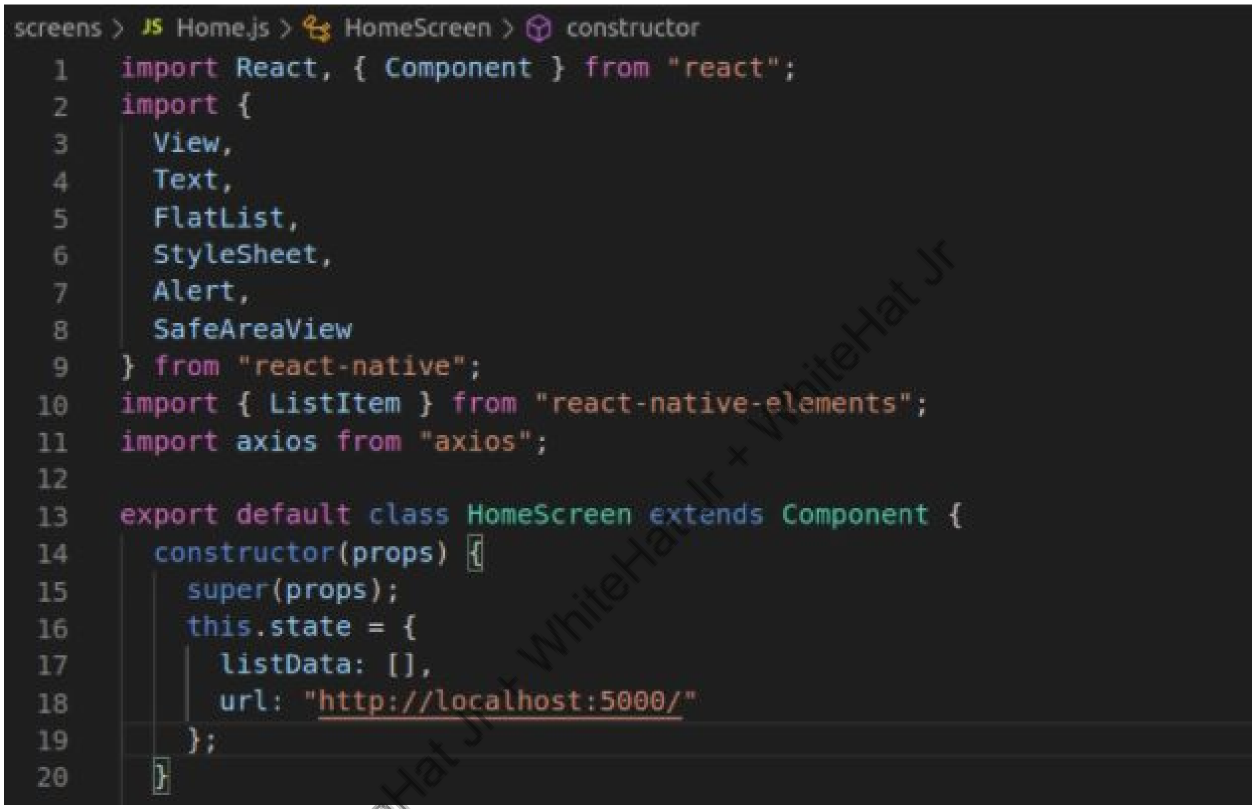


App.js > ...

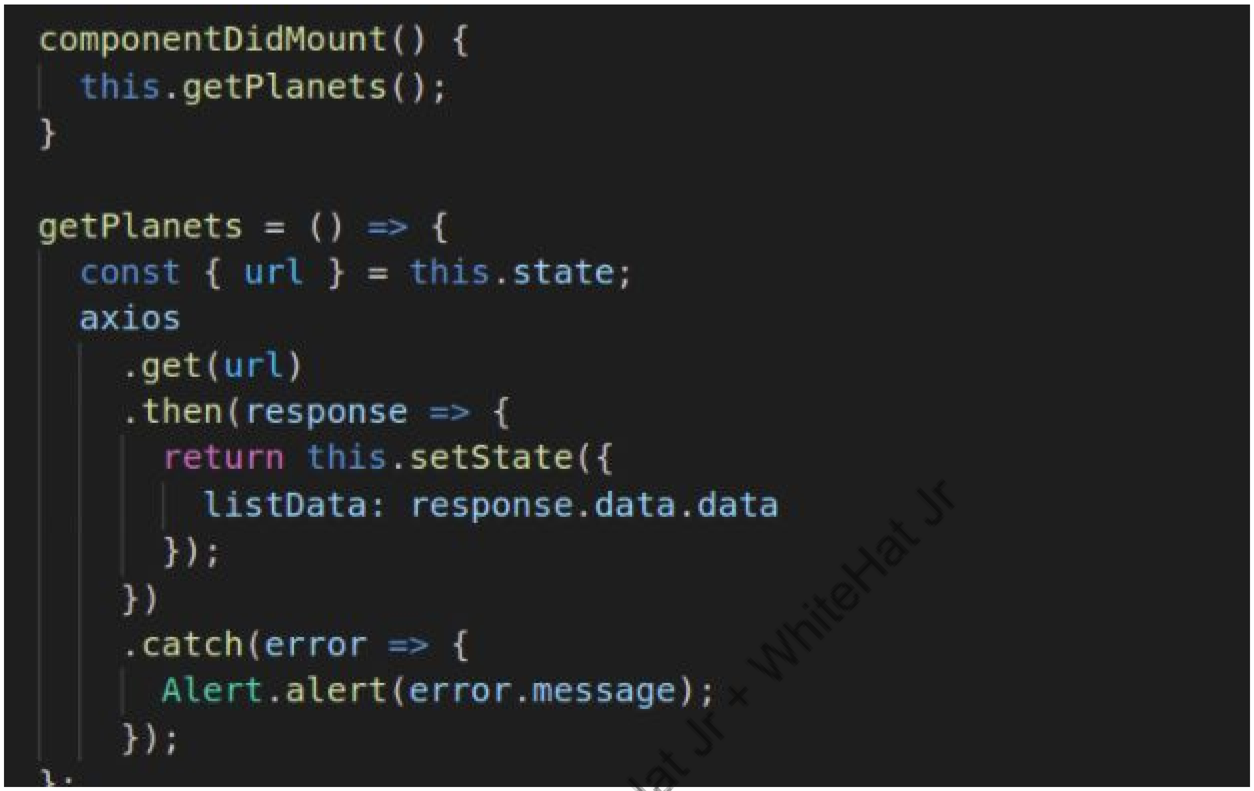
```
import React from "react";
import { createAppContainer } from "react-navigation";
import { createStackNavigator } from "react-navigation-stack";
import HomeScreen from "./screens/Home";
import DetailsScreen from "./screens/Details";

export default function App() {
  return <AppContainer />;
}

const appStackNavigator = createStackNavigator(
  {
    Home: {
      screen: HomeScreen,
      navigationOptions: {
        headerShown: false
      }
    },
    Details: {
      screen: DetailsScreen
    }
  },
  {
    initialRouteName: "Home"
  }
);

const AppContainer = createAppContainer(appStackNavigator);
```

8. Install axios so that we can make a query to our Flask API. We also need ListItems from react-native-elements.

   npm install axios
   npm install react-native-elements

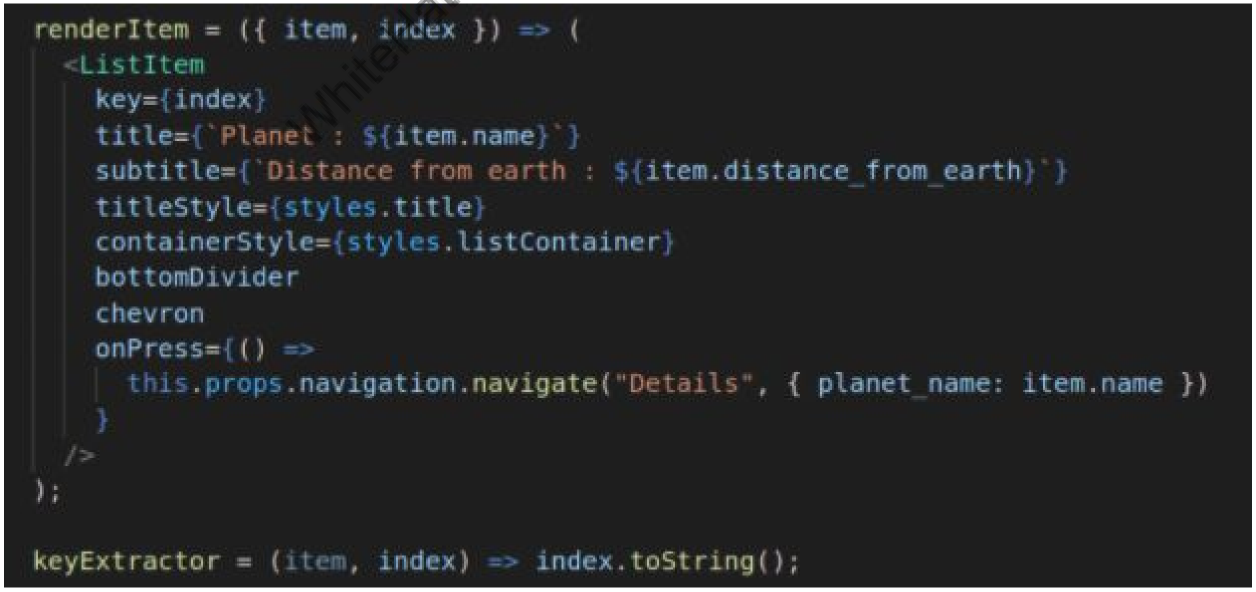


```
screens > JS Home.js > HomeScreen > constructor
import React, { Component } from "react";
import {
  View,
  Text,
  FlatList,
  StyleSheet,
  Alert,
  SafeAreaView
} from "react-native";
import { ListItem } from "react-native-elements";
import axios from "axios";

export default class HomeScreen extends Component {
  constructor(props) {
    super(props);
    this.state = {
      listData: [],
      url: "http://localhost:5000/"
    };
  }
```

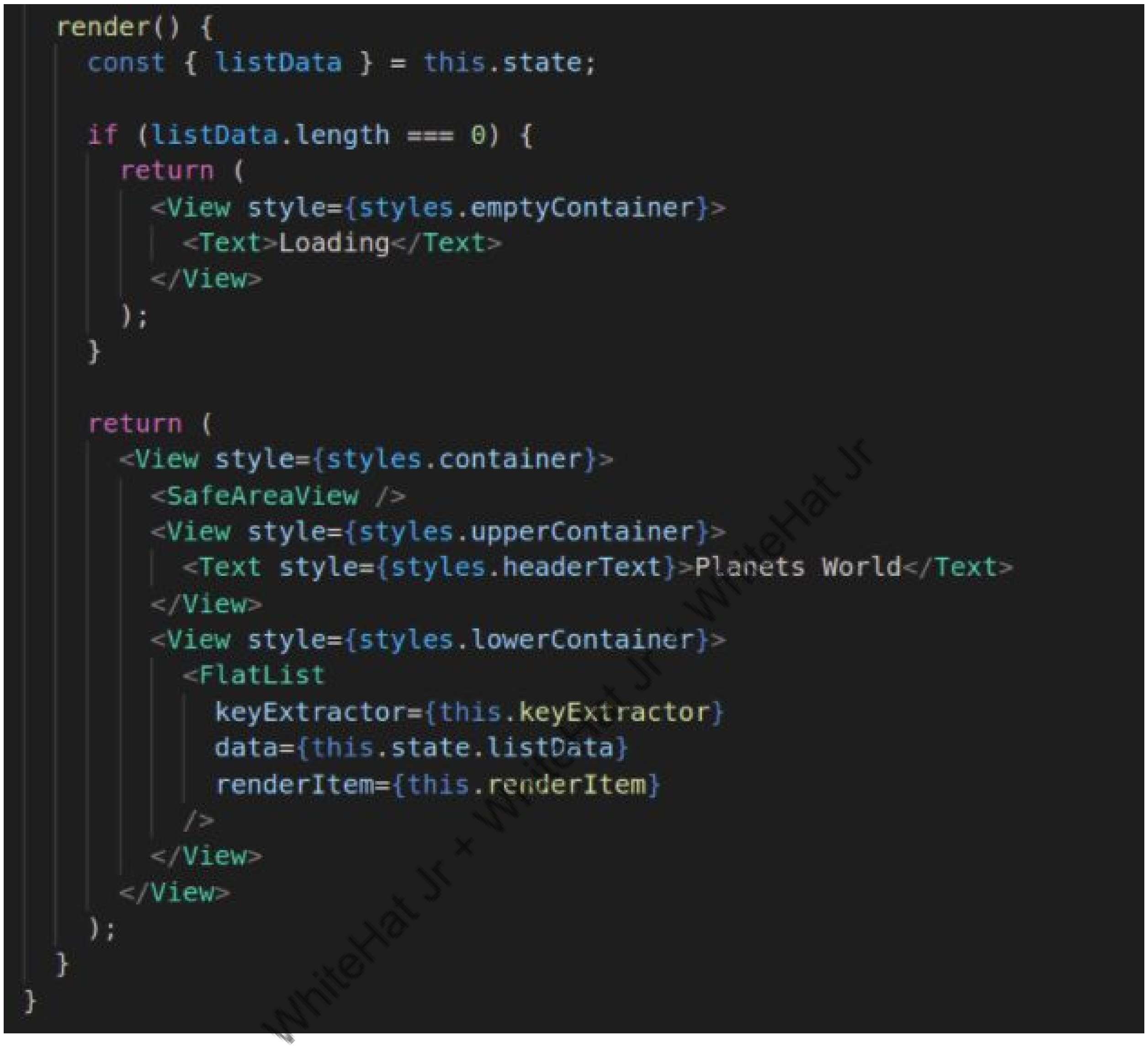


```
componentDidMount() {
  this.getPlanets();
}

getPlanets = () => {
  const { url } = this.state;
  axios
    .get(url)
    .then(response => {
      return this.setState({
        listData: response.data.data
      });
    })
    .catch(error => {
      Alert.alert(error.message);
    });
};
```

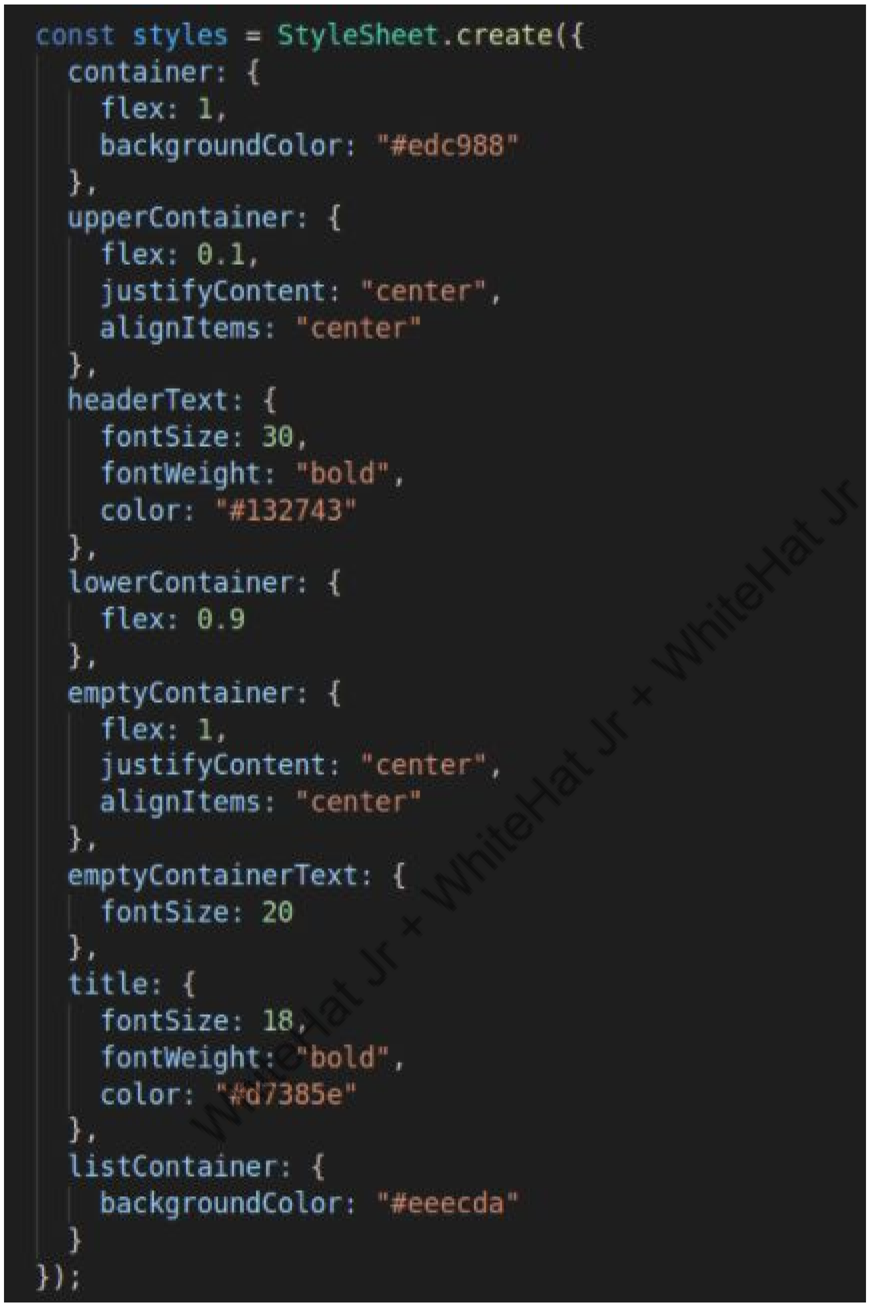
9. Display the data using FlatList. When the user presses on an item, we want to navigate the user to the Details Screen. We also need to make sure we pass the planet's name to the Details Screen. Remember how our Flask API used to work?



```
renderItem = ({ item, index }) => (
  <ListItem
    key={index}
    title={`Planet : ${item.name}`}
    subtitle={`Distance from earth : ${item.distance_from_earth}`}
    titleStyle={styles.title}
    containerStyle={styles.listContainer}
    bottomDivider
    chevron
    onPress={() =>
      this.props.navigation.navigate("Details", { planet_name: item.name })
    }
  />
);

keyExtractor = (item, index) => index.toString();
```

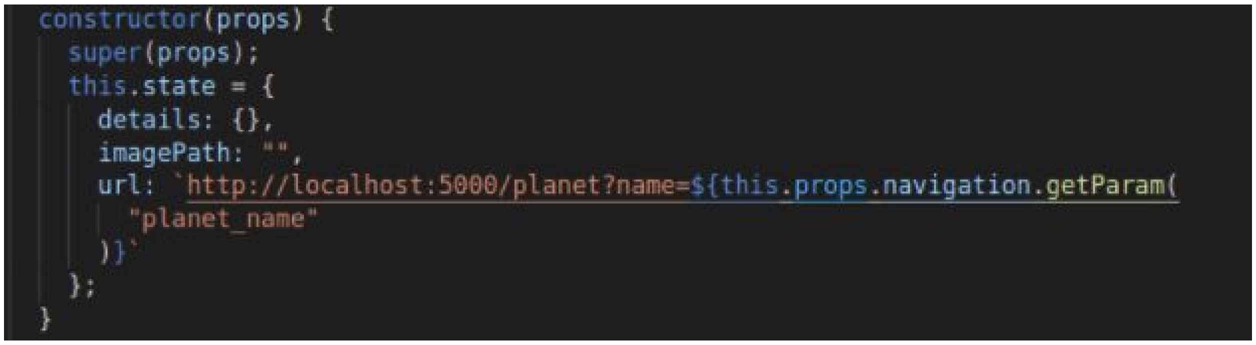


```
  render() {
    const { listData } = this.state;

    if (listData.length === 0) {
      return (
        <View style={styles.emptyContainer}>
          <Text>Loading</Text>
        </View>
      );
    }

    return (
      <View style={styles.container}>
        <SafeAreaView />
        <View style={styles.upperContainer}>
          <Text style={styles.headerText}>Planets World</Text>
        </View>
        <View style={styles.lowerContainer}>
          <FlatList
            keyExtractor={this.keyExtractor}
            data={this.state.listData}
            renderItem={this.renderItem}
          />
        </View>
      </View>
    );
  }
}
```



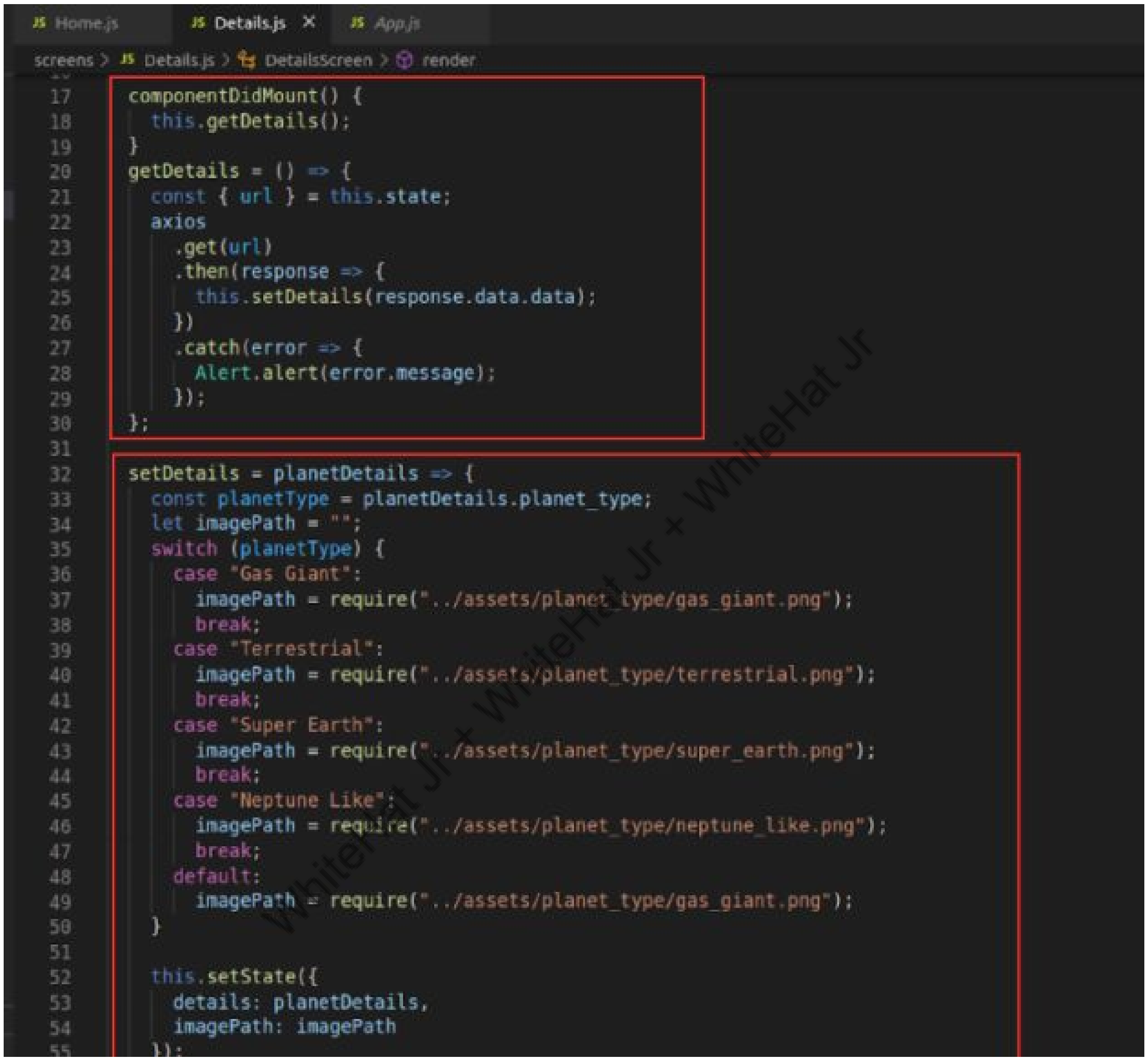
```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: "#edc988"
  },
  upperContainer: {
    flex: 0.1,
    justifyContent: "center",
    alignItems: "center"
  },
  headerText: {
    fontSize: 30,
    fontWeight: "bold",
    color: "#132743"
  },
  lowerContainer: {
    flex: 0.9
  },
  emptyContainer: {
    flex: 1,
    justifyContent: "center",
    alignItems: "center"
  },
  emptyContainerText: {
    fontSize: 20
  },
  title: {
    fontSize: 18,
    fontWeight: "bold",
    color: "#d7385e"
  },
  listContainer: {
    backgroundColor: "#eeecda"
  }
});
```

10. First, we'll create a state which will contain the details of the planet, imagePath of the planet and the url of the site.



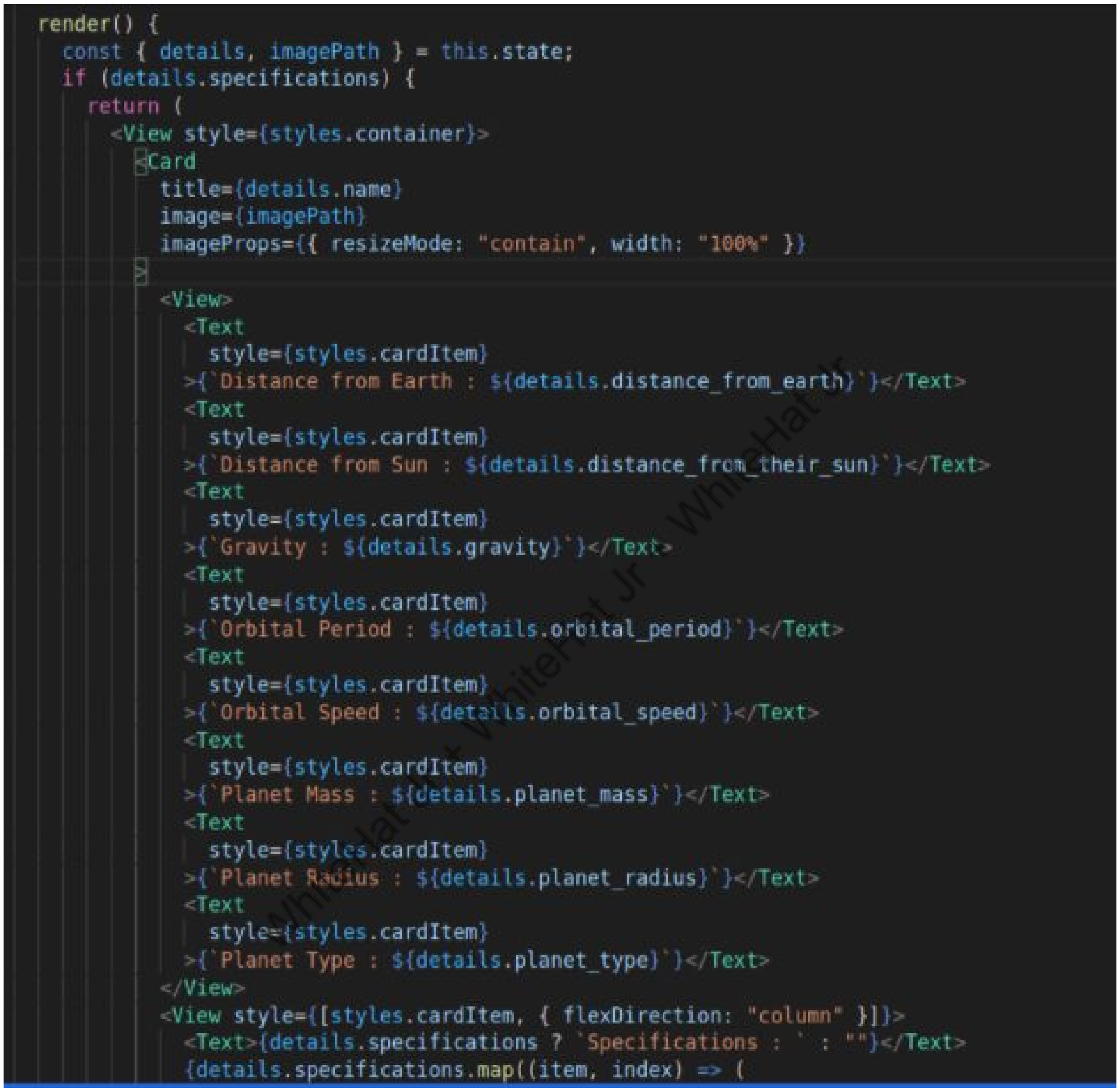
```
constructor(props) {
  super(props);
  this.state = {
    details: {},
    imagePath: "",
    url: `http://localhost:5000/planet?name=${this.props.navigation.getParam(
      "planet_name"
    )}`
  };
}
```

11. Create a **getDetails** function which will GET the data and give it to **setDetails** function.

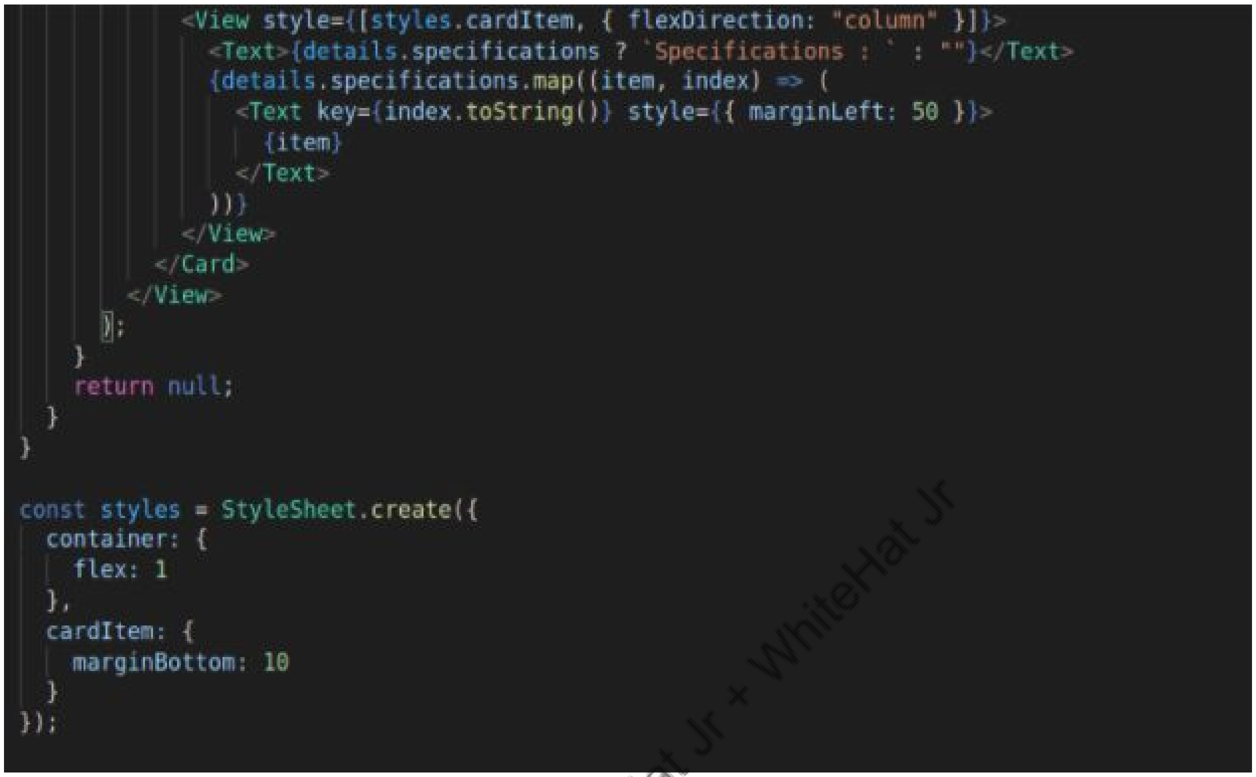


```
componentDidMount() {
  this.getDetails();
}
getDetails = () => {
  const { url } = this.state;
  axios
    .get(url)
    .then(response => {
      this.setDetails(response.data.data);
    })
    .catch(error => {
      Alert.alert(error.message);
    });
};

setDetails = planetDetails => {
  const planetType = planetDetails.planet_type;
  let imagePath = "";
  switch (planetType) {
    case "Gas Giant":
      imagePath = require("../assets/planet_type/gas_giant.png");
      break;
    case "Terrestrial":
      imagePath = require("../assets/planet_type/terrestrial.png");
      break;
    case "Super Earth":
      imagePath = require("../assets/planet_type/super_earth.png");
      break;
    case "Neptune Like":
      imagePath = require("../assets/planet_type/neptune_like.png");
      break;
    default:
      imagePath = require("../assets/planet_type/gas_giant.png");
  }

  this.setState({
    details: planetDetails,
    imagePath: imagePath
  });
```

12. To display all this information we'll use the cards from the react-native-elements.



```
render() {
  const { details, imagePath } = this.state;
  if (details.specifications) {
    return (
      <View style={styles.container}>
        <Card
          title={details.name}
          image={imagePath}
          imageProps={{ resizeMode: "contain", width: "100%" }}
        >
          <View>
            <Text
              style={styles.cardItem}
            >{`Distance from Earth : ${details.distance_from_earth}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Distance from Sun : ${details.distance_from_their_sun}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Gravity : ${details.gravity}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Orbital Period : ${details.orbital_period}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Orbital Speed : ${details.orbital_speed}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Planet Mass : ${details.planet_mass}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Planet Radius : ${details.planet_radius}`}</Text>
            <Text
              style={styles.cardItem}
            >{`Planet Type : ${details.planet_type}`}</Text>
          </View>
          <View style={[styles.cardItem, { flexDirection: "column" }]}>
            <Text>{details.specifications ? `Specifications : ` : ""}</Text>
            {details.specifications.map((item, index) => (
```



```
            <View style={[styles.cardItem, { flexDirection: "column" }]}>
              <Text>{details.specifications ? `Specifications : ` : ""}</Text>
              {details.specifications.map((item, index) => (
                <Text key={index.toString()} style={{ marginLeft: 50 }}>
                  {item}
                </Text>
              ))}
            </View>
          </Card>
        </View>
      );
    }
    return null;
  }
}

const styles = StyleSheet.create({
  container: {
    flex: 1
  },
  cardItem: {
    marginBottom: 10
  }
});
```

13. The code is ready! Run, test and debug the code.

What's NEXT?

In the next class, we will start with our Capstone Project!

EXTEND YOUR KNOWLEDGE:

You can read the following blog on speed of our planet to understand more:
https://reactnativeelements.com/docs/listitem