Een gyroscoop met een bewegend object en een grafiek.  
  
Deze dagen is een gyroscoop systeem voor spellen of apps in het algemeen. De app moet bij kunnen houden hoe de telefoon gedraaid is en daarbij het scherm juist te kunnen plaatsten. We laten dus deze app de gyroscoop in je telefoon lezen en dit laten we in nummers en een grafiek zien.   
  
We lever je een klein zipbestand  
om het simpel te houden  
  
Voorbereiding  
pak uit SetUp.Zip  
  
Run makeit.cmd  
  
Geef het een naam.  
  
Maak een Blank Template aan.  
  
Open het in visual studio code.  
  
Uitvoering:

1. Open een terminal en typ   
     
   “npm install”
2. Ga naar …/components/Detailscreen  
     
   Ga naar je terminal en vul in  
   “expo install expo-sensors”
3. Plak dit in Homescreen in je homescreen component

“import React, { useState, useEffect } from "react";

import {

  StyleSheet,

  Text,

  TouchableOpacity,

  View, Dimensions,

} from "react-native";

import { Gyroscope } from "expo-sensors";

// import { BarChart } from "react-native-gifted-charts";

import {

  PieChart,

  LineChart,

} from 'react-native-chart-kit';

const HomeScreen = () => {

  const [data, setData] = useState({

    x: 1,

    y: 1,

    z: 1,

  });

  const [subscription, setSubscription] = useState(null);

  const \_slow = () => {

    Gyroscope.setUpdateInterval(1000);

  };

  const \_fast = () => {

    Gyroscope.setUpdateInterval(500);

  };

  const { x, y, z } = data;

  const \_subscribe = () => {

    setSubscription(

      Gyroscope.addListener((gyroscopeData) => {

        setData(gyroscopeData);

      })

    );

  };

  const \_unsubscribe = () => {

    subscription && subscription.remove();

    setSubscription(null);

  };

  useEffect(() => {

    \_subscribe();

    return () => \_unsubscribe();

  }, []);

  return (

    <View style={styles.container}>

      <Text style={styles.text}>Gyroscope:</Text>

      <>

        <Text style={styles.header}>Bezier Line Chart</Text>

        <LineChart

          data={{

            labels: [],

            datasets: [

              {

                data: [

                  data.x,

                  data.y,

                  data.z,

                  data.x,

                  data.y,

                  data.z,

                ]

              },

              {

                data: [1], //highest graph value

                withDots: false, //a flage to make it hidden

              },

              {

                data: [0], //highest graph value

                withDots: false, //a flage to make it hidden

              },

              {

                data: [-1], //highest graph value

                withDots: false, //a flage to make it hidden

              },

            ],

          }}

          width={Dimensions.get('window').width - 16} // from react-native

          height={220}

          // yAxisLabel={'Rs'}

          chartConfig={{

            backgroundColor: '#1cc910',

            backgroundGradientFrom: '#eff3ff',

            backgroundGradientTo: '#efefef',

            decimalPlaces: 0, // optional, defaults to 2dp

            color: (opacity = 255) => `rgba(0, 0, 0, ${opacity})`,

            style: {

              borderRadius: 16,

            },

          }}

          // bezier

          style={{

            marginVertical: 8,

            borderRadius: 16,

          }}

        />

      </>

      <Text style={styles.text}>

        x: {round(x)} y: {round(y)} z: {round(z)}

      </Text>

      <View style={styles.buttonContainer}>

        <TouchableOpacity

          onPress={subscription ? \_unsubscribe : \_subscribe}

          style={styles.button}

        >

          <Text>{subscription ? "On" : "Off"}</Text>

        </TouchableOpacity>

        <TouchableOpacity

          onPress={\_slow}

          style={[styles.button, styles.middleButton]}

        >

          <Text>Slow</Text>

        </TouchableOpacity>

        <TouchableOpacity onPress={\_fast} style={styles.button}>

          <Text>Fast</Text>

        </TouchableOpacity>

      </View>

    </View>

    //     <>

    //       <Text style={styles.header}>Line Chart</Text>

    //       <LineChart

    //         data={{

    //           labels: ['January', 'February', 'March'],

    //           datasets: [

    //             {

    //               data: [date1, date2, date3],

    //               strokeWidth: 2,

    //             },

    //           ],

    //         }}

    //         width={Dimensions.get('window').width - 16}

    //         height={220}

    //         chartConfig={{

    //           backgroundColor: '#1cc910',

    //           backgroundGradientFrom: '#eff3ff',

    //           backgroundGradientTo: '#efefef',

    //           decimalPlaces: 2,

    //           color: (opacity = 1) => `rgba(0, 0, 0, ${opacity})`,

    //           style: {

    //             borderRadius: 16,

    //           },

    //         }}

    //         style={{

    //           marginVertical: 8,

    //           borderRadius: 16,

    //         }}

    //       />

    //     </>

    //   );

    // };

  );

};

function round(n) {

  if (!n) {

    return 0;

  }

  return Math.floor(n \* 100) / 100;

}

export default HomeScreen;

const styles = StyleSheet.create({

  container: {

    flex: 1,

    justifyContent: "center",

    paddingHorizontal: 10,

  },

  text: {

    textAlign: "center",

  },

  buttonContainer: {

    flexDirection: "row",

    alignItems: "stretch",

    marginTop: 15,

  },

  button: {

    flex: 1,

    justifyContent: "center",

    alignItems: "center",

    backgroundColor: "#eee",

    padding: 10,

  },

  middleButton: {

    borderLeftWidth: 1,

    borderRightWidth: 1,

    borderColor: "#ccc",

  },

});

4. Voeg onderstaande code toe aan het component listscreen

import { StyleSheet, Text, View, TextInput, Dimensions, Button, FlatList, Alert } from 'react-native'

import React, { useState } from 'react';

import {

    LineChart, BarChart

} from 'react-native-chart-kit';

const ListScreen = (props) => {

    console.log(props);

    console.log(props.route.params);

    const average = props.route.params.average;

    const Setstudenten = props.route.params.student;

    console.log(average);

    console.log(Setstudenten);

    const labels = Setstudenten.map(item => {

        return item.naam;

    });

    const values = Setstudenten.map(item => {

        return item.score;

    });

    console.log(values);

    console.log(labels);

    return (

        <View style={styles.container}>

            <LineChart

                data={{

                    labels: labels,

                    datasets: [

                        {

                            data: values,

                        },

                        {

                            data: [average], //highest graph value

                            withDots: true, //a flage to make it hidden

                            color: (opacity = 5) => `rgba(25, 150, 25, ${opacity})`,

                            strokeWidth: 4,

                            strokeColor: "#ff0000",

                            fillColor: "#ff0000",

                            fillAlpha: 0.5,

                        },

                        {

                            data: [10], //highest graph value

                            withDots: false, //a flage to make it hidden

                        },

                        {

                            data: [average], //highest graph value

                            withDots: false, //a flage to make it hidden

                        },

                        {

                            data: [0], //highest graph value

                            withDots: false, //a flage to make it hidden

                        },

                    ],

                }}

                width={Dimensions.get("window").width - 16} // from react-native

                height={220}

                // yAxisLabel={'Rs'}

                chartConfig={{

                    backgroundColor: "#1cc910",

                    backgroundGradientFrom: "#eff3ff",

                    backgroundGradientTo: "#efefef",

                    decimalPlaces: 0, // optional, defaults to 2dp

                    color: (opacity = 255) => `rgba(0, 0, 0, ${opacity})`,

                    style: {

                        borderRadius: 16,

                    },

                }}

                // bezier

                style={{

                    marginVertical: 8,

                    borderRadius: 16,

                }}

            />

            <Text> het gemiddelde is: {average}</Text>

        </View>

    );

}

export default ListScreen

const styles = StyleSheet.create({

    container: {

        flex: 1,

        backgroundColor: '#fff',

        alignItems: 'center',

        justifyContent: 'center',

    },

});

5. Voeg onderstaande code toen aan het component detailscreen

import { StyleSheet, Text, View, TextInput, Button, FlatList, Alert } from 'react-native'

import React, { useState } from 'react';

const DetailScreen = (props) => {

    const [naam, setname] = useState('');

    const [score, setscore] = useState(0);

    const [id, setid] = useState(1);

    const [student, Setstudent] = useState([]);

    const del = (item) => {

        Setstudent(student.filter(items => items.id !== item.id));

    }

    const values = student.map(item => {

        return item.score;

    });

    const onChangeName = (value) => {

        setname(value);

    };

    const onChangescore = (value) => {

        if (value < 11) {

            setscore(value);

        }

        else {

            alert("score mag niet hoger dan 10 zijn");

            Alert.alert("score mag niet hoger dan 10 zijn");

            setscore(0);

        }

    };

    const make = () => {

        const average = values.reduce((a, b) => a + b) / values.length;

        props.navigation.push('stList', { student: student, average: average });

    }

    const add = () => {

        Setstudent([...student, { id: id, naam: naam, score: Number(score) }]);

        setid(id + 1);

        setname('');

        setscore(0);

    }

    const item = (item) => {

        return (

            <View style={{ flexDirection: 'row' }}>

                <View style={{ width: 100, backgroundColor: 'red' }}>

                    <Text >{item.item.naam}</Text>

                </View>

                <View style={{ width: 100, backgroundColor: 'red' }}>

                    <Text>{item.item.score}</Text>

                </View>

                <View style={{ width: 100, backgroundColor: 'red' }}>

                    <Button title="Del" onPress={() => del(item.item)} />

                </View>

            </View>

        );

    }

    return (

        <View style={styles.container}>

            <View style={{ flexDirection: "row" }}>

                <View style={{ width: 100, backgroundColor: "red" }}>

                    <Text style={{ width: 100, color: "black" }}>name</Text>

                </View>

                <View style={{ width: 100, backgroundColor: "red" }}>

                    <Text>params</Text>

                </View>

                <View style={{ width: 100, backgroundColor: "red" }}></View>

            </View>

            <FlatList

                data={student}

                renderItem={item}

                keyExtractor={(item, index) => index.toString()}

            />

            <View style={styles.container}>

                <View style={{ flexDirection: "row" }}>

                    <Text style={{ width: 100, color: "black" }}>student naam</Text>

                    <TextInput

                        style={styles.inpout}

                        value={naam}

                        onChangeText={(value) => onChangeName(value)}

                    />

                </View>

                <View style={{ flexDirection: "row" }}>

                    <Text style={{ width: 100, color: "black" }}>score</Text>

                    <TextInput

                        style={styles.inpout}

                        value={score}

                        onChangeText={(value) => onChangescore(value)}

                    />

                </View>

                <View style={{ width: 200, backgroundColor: "red" }}>

                    <Button title="Add to table" onPress={() => add()} />

                </View>

                <View style={{ width: 200, backgroundColor: "red" }}>

                    <Button title="Make chart" onPress={() => make()} />

                </View>

            </View>

        </View>

    );

}

export default DetailScreen

const styles = StyleSheet.create({

    container: {

        flex: 1,

        backgroundColor: '#fff',

        alignItems: 'center',

        justifyContent: 'center',

        marginTop: '10%',

    },

    container1: {

        flex: 1,

        backgroundColor: '#fff',

        alignItems: 'center',

        justifyContent: 'center',

        marginBottom: '50% ',

    },

    inpout: {

        width: 100,

        height: 40,

        backgroundColor: '#7f7f',

        background: '#7f7f',

        border: '4px solid #7f7f',

    },

});

6. Maak een component aan genaamd CompseScreen (voeg de route ook aan app.js toe.)  
  
7. Voeg onderstaande code toe aan het component CompseScreen.

import React, { useState, useEffect } from "react";

import { Image, View, Text, Dimensions } from "react-native";

import { Grid, Col, Row } from "react-native-easy-grid";

import { Magnetometer } from "expo-sensors";

const { height, width } = Dimensions.get("window");

const ComposeScreen = () => {

    const [subscription, setSubscription] = useState(null);

    const [magnetometer, setMagnetometer] = useState(0);

    useEffect(() => {

        \_toggle();

        return () => {

            \_unsubscribe();

        };

    }, []);

    const \_toggle = () => {

        if (subscription) {

            \_unsubscribe();

        } else {

            \_subscribe();

        }

    };

    const \_subscribe = () => {

        setSubscription(

            Magnetometer.addListener((data) => {

                setMagnetometer(\_angle(data));

            })

        );

    };

    const \_unsubscribe = () => {

        subscription && subscription.remove();

        setSubscription(null);

    };

    const \_angle = (magnetometer) => {

        let angle = 0;

        if (magnetometer) {

            let { x, y, z } = magnetometer;

            if (Math.atan2(y, x) >= 0) {

                angle = Math.atan2(y, x) \* (180 / Math.PI);

            } else {

                angle = (Math.atan2(y, x) + 2 \* Math.PI) \* (180 / Math.PI);

            }

        }

        return Math.round(angle);

    };

    const \_direction = (degree) => {

        if (degree >= 22.5 && degree < 67.5) {

            return "NE";

        } else if (degree >= 67.5 && degree < 112.5) {

            return "E";

        } else if (degree >= 112.5 && degree < 157.5) {

            return "SE";

        } else if (degree >= 157.5 && degree < 202.5) {

            return "S";

        } else if (degree >= 202.5 && degree < 247.5) {

            return "SW";

        } else if (degree >= 247.5 && degree < 292.5) {

            return "W";

        } else if (degree >= 292.5 && degree < 337.5) {

            return "NW";

        } else {

            return "N";

        }

    };

    // Match the device top with pointer 0° degree. (By default 0° starts from the right of the device.)

    const \_degree = (magnetometer) => {

        return magnetometer - 90 >= 0 ? magnetometer - 90 : magnetometer + 271;

    };

    return (

        <Grid style={{ backgroundColor: "black" }}>

            <Row style={{ alignItems: "center" }} size={0.9}>

                <Col style={{ alignItems: "center" }}>

                    <Text

                        style={{

                            color: "#fff",

                            fontSize: height / 26,

                            fontWeight: "bold",

                        }}

                    >

                        {\_direction(\_degree(magnetometer))}

                    </Text>

                </Col>

            </Row>

            <Row style={{ alignItems: "center" }} size={0.1}>

                <Col style={{ alignItems: "center" }}>

                    <View

                        style={{

                            position: "absolute",

                            width: width,

                            alignItems: "center",

                            top: 0,

                        }}

                    >

                        <Image

                            source={require("../assets/compass\_pointer.png")}

                            style={{

                                height: height / 26,

                                resizeMode: "contain",

                            }}

                        />

                    </View>

                </Col>

            </Row>

            <Row style={{ alignItems: "center" }} size={2}>

                <Text

                    style={{

                        color: "#fff",

                        fontSize: height / 27,

                        width: width,

                        position: "absolute",

                        textAlign: "center",

                    }}

                >

                    {\_degree(magnetometer)}°

                </Text>

                <Col style={{ alignItems: "center" }}>

                    <Image

                        source={require("../assets/compass\_bg.png")}

                        style={{

                            height: width - 80,

                            justifyContent: "center",

                            alignItems: "center",

                            resizeMode: "contain",

                            transform: [{ rotate: 360 - magnetometer + "deg" }],

                        }}

                    />

                </Col>

            </Row>

            <Row style={{ alignItems: "center" }} size={1}>

                <Col style={{ alignItems: "center" }}>

                    <Text style={{ color: "#fff" }}>Copyright @HamzahSari</Text>

                </Col>

            </Row>

        </Grid>

    );

};

export default ComposeScreen;

8. Pak de assets die je nodig hebt van de github.