**React ChartJs Info**

1. In this documentation, We will be talking about the world famous library called the chartJs
2. For making the use of this library, We first have to install the following packages in the terminal of the project,

**npm install react-chartjs-2 chart.js**

1. In this example we will take the data from the file named the Data.js, but take care that the data you use should be good with the cart, as the some data does not work good with the charts
2. We, will show a graph of showing the growth of the user in the app
3. For any of the graph, chart we make by the help of the react-chartjs-2 package, we have to import the following statements,(In this example we have make the new file named the <name>Graph.js and in it, We will follow the following instructions)

🡪For BarChart

**import { Bar } from “react-chartjs-2”;**

**import { Chart as ChartJS } from “chart.js/auto”**

🡪For LineChart

**import { Line } from “react-chartjs-2”;**

**import { Chart as ChartJS} from “chart.js/auto”**

🡪For PieChart

**import { Pie } from “react-chartjs-2”;**

**import { Chart as ChartJS } from “chart.js/auto”;**

1. And after doing this imports, we can easily use the alias of the imported item, example **<Bar/>, <Line/>, <Pie/>**
2. And in the alias, we can also give the two props, the one is data and the other is options, But the options can be neglected for now, The genric form of the statement is :

**<Bar data={} options={}/>;**

1. And as told earlier when we need the data we should pass it via props or any other thing, For this example we will pass it by the help of the props
2. And, After completing the following steps, Now we will make this alias of the graph to be used in the App.js and init we will pass the data as the porps

**<BarChart chartData={}/>**

1. Now to pass the data in the props, we will make an useState const, Example

**const [userData, setUserData]=useState({**

**labels: “”,**

**datasets: “”**

**})**

🡪And in the labels, We have to put the labels of the data which is in the array of the data, and this will be used in the graph, Example in this example it is years 2016, 2017…

🡪But it is not efficient way to type all the labels,As we have lot of the data

Eg. **labels: [2016, 2017],**

🡪Instead we can follow the following method with the map function, Example

**labels: UserData.map((data)=>data.year),**

1. And than in the useState after the labels, there comes datasets, In it we have to give all the information about the data, And in it’s label we have to give the main thing about which our data is of
2. And than comes the main part of the data, which is inside the datasets and after the label, And in it we have the two options, The one is to hard code the value and the other is to make the value be used by the help of the map function
3. And by this things only we can get the data in the form of the chart, But the chart would be in the black and white colour which would not look nice

🡪And we can make it colouring by the help of the backgroundColor and in the input it would require an array colour for each column

🡪If we would give the only one colour than it will be implemented to all the columns

🡪If we will give the two colors than the colours will make work alternatively

1. We can also make the size of the chart to be small, By covring the alias in the App.js by the div’s and containers, and CSS width, Example

**<div className=”container” style={{width: 200}}>**

**<BarChart chartData={userData}/>**

**</div>**

1. We can also add the border to the columns of the chats
2. The sample code of the above project is as follow:

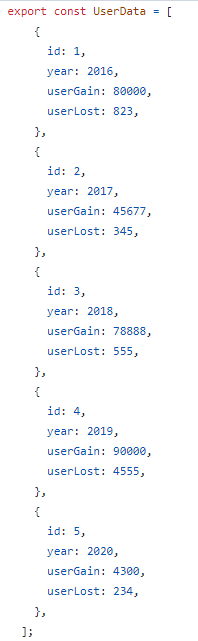


PieChart.js/components

BarChart.js/components

LineChart.js/components

App.js



Data.js

1. We can also make the compare the data of the more than two products in the bar chart and the line chart but not in the pie chat
2. For comparing more than two datas, we have to make the two arrays of the data with the different names
3. And than in the App.js, we have to make no change in the alias that we are sending, but we have to add the Another data in the datasets and in it we have to make some difference in the color, label and data name, Else every thing will remain same
4. The example of the datasets in the App.js is as follow:



The example of the upper part of the App.js