Project Development Phase

Sprint – 3

Team ID	PNT2022TMID30856
Project Name	Personal Expense Tracker Application
Batch Number	B8-2A4E

Navbar.jsx

```
import React from 'react';
import { useNavigate } from 'react-router';
import "./navbar.css";

const Navbar = () => {

  const navigate = useNavigate();

  const handleLogout = () => {
    navigate('/');
  }

  return (
    <div className="navbar">
      <div className="navTitle">Personal Expense Tracker
        <span className='navSubTitle'>speak to track</span>
      </div>
      <div>
        <button className='navBtn' onClick={() => handleLogout()}>Logout</button>
      </div>

    </div>
```

```
    );
}


export default Navbar;
```

navbar.css

```
.navbar {
    display: flex;
    flex-direction: row;
    height: 70px;
    background-color: white;
    border: 3px solid darkgoldenrod;
    border-radius: 5px;
    margin-top: 10px;
    align-items: center;
    justify-content: space-between;

}

.navTitle {
    font-size: 30px;
    margin-left: 15%;
    text-align: center;
    color: rgb(187, 134, 0);
    font-weight: bold;
    letter-spacing: 1px;
}

.navSubTitle {
```

```
    letter-spacing: normal;
    font-size: 20px;
    color: grey;
    margin-left: 75px;
}

.navBtn {
    height: 70%;
    margin-right: 220px;
    background-color: rgb(228, 227, 194);
    border-radius: 5px;
    border: 2px solid rgb(111, 95, 0);
    font-weight: 600;
    color: rgb(194, 128, 6);
    padding: 10px 20px;
    cursor: pointer;
}
```

Details.jsx

```
import React from 'react';
import { Card, CardContent } from '@mui/material';
import { Doughnut, Pie, PolarArea, Radar } from 'react-chartjs-2';
import './details.css';
import useTransactions from '../../useTransactions';
import 'chart.js/auto';

const Details = ({ title }) => {

    /*const [doughnatC, setDoughnatC] = useState(true);
    const [polarAreaC, setPolarAreaC] = useState(false);
```

```
  const [pieC, setPieC] = useState(false);
  const [radarC, setRadarC] = useState(false);*/

  const { chartData } = useTransactions(title);

  console.log(chartData);

  return (
    <div>
      <div style={{ display: 'flex', flexDirection: 'row', gap: '5px', marginTop: '10px' }}>

      <Card style={{}} className={title === 'Income' ? "income" : "expense"} >
      {/*<div className="chartButtonContainer" >
        <button className={title === 'Income' ? (doughnatC ? "selectedIn" : "btnIn") :
(doughnatC ? "selectedEx" : "btnEx") }
          onClick={() => {
            setDoughnatC(true);
            setPolarAreaC(false);
            setPieC(false);
            setRadarC(false);
          }}
        >Doughnat</button>
        <button className={title === 'Income' ? (polarAreaC ? "selectedIn" : "btnIn") :
(polarAreaC ? "selectedEx" : "btnEx")}
          onClick={() => {
            setDoughnatC(false);
            setPolarAreaC(true);
            setPieC(false);
            setRadarC(false);
          }}
        >PolarArea</button>
```

```
            <button className={title === 'Income' ? (pieC ? "selectedIn" :"btnIn") : (pieC ? "selectedEx" : "btnEx")}
                onClick={() => {
                    setDoughnatC(false);
                    setPolarAreaC(false);
                    setPieC(true);
                    setRadarC(false);
                }}
            >Pie</button>
            <button className={title === 'Income' ? (radarC ? "selectedIn" : "btnIn") : (radarC ? "selectedEx" : "btnEx")}
                onClick={() => {
                    setDoughnatC(false);
                    setPolarAreaC(false);
                    setPieC(false);
                    setRadarC(true);
                }}
            >Radar</button>
        </div>*/}
            {/*<CardHeader style={{ textAlign: "center", }} title={title+": Rs. "+total} />*/}
            <CardContent>
                <Doughnut data={chartData} />
            </CardContent>
        </Card>

        <Card className={title === 'Income' ? "income" : "expense"} >
            <CardContent>
                <PolarArea data={chartData} />
            </CardContent>
        </Card>
        <Card className={title === 'Income' ? "income" : "expense"} >
```

```
            <CardContent>
              <Pie data={chartData} />
            </CardContent>
          </Card>
          <Card className={title === 'Income' ? "income" : "expense"} >
            <CardContent>
              <Radar data={chartData} />
            </CardContent>
          </Card>



            {/*{doughnatC &&
              <Doughnut data={chartData} />
            }
            {polarAreaC &&
              <PolarArea data={chartData} />
            }
            {pieC &&
              <Pie data={chartData} />
            }
            {radarC &&
              <Radar data={chartData} />
            }*/}
          </div>
        </div>

    );
};

export default Details;
```

details.css

```
.income {
   border-top: 10px solid rgba(0, 255, 0, 0.7);
   border-bottom: 10px solid rgba(0, 255, 0, 0.7);
}

.expense {
   border-top: 10px solid rgba(255, 0, 0, 0.7);
   border-bottom: 10px solid rgba(255, 0, 0, 0.7);
}
```

useTransactions.js

```
import { useContext } from "react"
import { expenseCategories, incomeCategories, resetCategories } from
"./constants/categories";
import { ExpenseTrackerContext } from "./context/context"



const useTransactions = (title) => {
   resetCategories();
   const { transactions } = useContext(ExpenseTrackerContext);
   const transactionsPerType = transactions.filter((t) => t.type === title);
   const total = transactionsPerType.reduce((acc, currVal) => acc += currVal.amount, 0);
   const categories = title === 'Income' ? incomeCategories : expenseCategories;
   var month = "";
   var monthIncomeTotal = [
      {m:"January", amount:0},
      {m:"February", amount:0},
      {m:"March", amount:0},
      {m:"April", amount:0},
      {m:"May", amount:0},
```

```
    {m:"June",  amount:0},
    {m:"July",  amount:0},
    {m:"August", amount:0},
    {m:"September", amount:0},
    {m:"October", amount:0},
    {m:"November",  amount:0},
    {m:"December",  amount:0},
];
var monthExpenseTotal = [
    {m:"January", amount:0},
    {m:"February", amount:0},
    {m:"March", amount:0},
    {m:"April", amount:0},
    {m:"May",  amount:0},
    {m:"June",  amount:0},
    {m:"July",  amount:0},
    {m:"August", amount:0},
    {m:"September", amount:0},
    {m:"October", amount:0},
    {m:"November",  amount:0},
    {m:"December",  amount:0},
];

// transactionsPerType.forEach((t) => {
//     console.log(t.amount, t.category, t.type);
// })

transactionsPerType.forEach((t) => {

    month = t.date.slice(5, 7);
```

```
if (t.type === 'Income') {
    switch(month) {
        case '01': monthIncomeTotal[0].amount += t.amount;break;
        case '02': monthIncomeTotal[1].amount += t.amount;break;
        case '03': monthIncomeTotal[2].amount += t.amount;break;
        case '04': monthIncomeTotal[3].amount += t.amount;break;
        case '05': monthIncomeTotal[4].amount += t.amount;break;
        case '06': monthIncomeTotal[5].amount += t.amount;break;
        case '07': monthIncomeTotal[6].amount += t.amount;break;
        case '08': monthIncomeTotal[7].amount += t.amount;break;
        case '09': monthIncomeTotal[8].amount += t.amount;break;
        case '10': monthIncomeTotal[9].amount += t.amount;break;
        case '11': monthIncomeTotal[10].amount += t.amount;break;
        case '12': monthIncomeTotal[11].amount += t.amount;break;
        default: break;
    }
}
else if(t.type === 'Expense') {
    switch(month) {
        case '01': monthExpenseTotal[0].amount += t.amount;break;
        case '02': monthExpenseTotal[1].amount += t.amount;break;
        case '03': monthExpenseTotal[2].amount += t.amount;break;
        case '04': monthExpenseTotal[3].amount += t.amount;break;
        case '05': monthExpenseTotal[4].amount += t.amount;break;
        case '06': monthExpenseTotal[5].amount += t.amount;break;
        case '07': monthExpenseTotal[6].amount += t.amount;break;
        case '08': monthExpenseTotal[7].amount += t.amount;break;
        case '09': monthExpenseTotal[8].amount += t.amount;break;
        case '10': monthExpenseTotal[9].amount += t.amount;break;
        case '11': monthExpenseTotal[10].amount += t.amount;break;
```

```
            case '12': monthExpenseTotal[11].amount += t.amount;break;
            default: break;
        }
    }


    const category = categories.find((c) => c.type === t.category);

    if (category) category.amount += t.amount;
});




const filteredCategories = categories.filter((c) => c.amount > 0);

const chartData = {
    datasets: [{
        data: filteredCategories.map((c) => c.amount),
        backgroundColor: filteredCategories.map((c) => c.color),
    }],
    labels: filteredCategories.map((c) => c.type)
};



const chartDataIncome = {
    datasets: [
        {
            label: 'Income',
```

```
            data: monthIncomeTotal.map((m) => m.amount),
            borderColor: '#165f40',
            backgroundColor: '#0bc77e',
            tension: 0.1,
          },

        ],
        labels: monthIncomeTotal.map((m) => m.m)
    };

    const chartDataExpense = {
        datasets: [
          {
            label: 'Expense',
            data: monthExpenseTotal.map((m) => m.amount),
            borderColor: '#b50d12',
            backgroundColor: '#e57c58',
            tension: 0.1,
          },
        ],
        labels: monthExpenseTotal.map((m) => m.m)
    };


    return { total, chartData, chartDataIncome, chartDataExpense };
};


export default useTransactions;
```

contextReducer.js

```
const contextReducer = (state, action) => {
    let transactions;

    switch (action.type) {
        case 'DELETE_TRANSACTION':
            transactions = state.filter((t) => t.id !== action.payload);

            localStorage.setItem('transaction',  JSON.stringify(transactions));

            return transactions;

        case 'ADD_TRANSACTION':
            transactions = [action.payload, ...state];

            localStorage.setItem('transaction',  JSON.stringify(transactions));

            return transactions;

        default:
            return state;
    }
}

export default contextReducer;
```

OUTPUT

Expense Graph: