

## ProjectDevelopmentPhas

### eSprint – 2

#### SourceCode

Date	04November2022
TeamID	PNT2022TMID23145
ProjectName	PersonalExpenseTracker
BatchNumber	B9-3A5E

#### Dashboard.jsx

```
import{PushToTalkButton,PushToTalkButtonContainer}from'@speechly/react-  
ui';importReactfrom'react';  
import{Details,Main}from'../components';  
importDetailsTrackfrom'../components/DetailsTrack/DetailsTra  
ck';importNavbarfrom'../components/Navbar/Navbar';
```

```
constDashboard=()=>{
```

```
  return(  
    <div>  
      <divstyle={{display:'flex',flexDirection:'row'}}>  
  
        <div>  
          <Main/>  
        </div>  
        <div>  
          <div>  
            <Navbar/>  
          </div>  
      </div>  
    </div>
```

```

        <span>
          <Detailstitle="Income"/>
        </span>
        <span>
          <Detailstitle="Expense"/>
        </span>
      </div>

      <PushToTalkButtonContainer>
        <PushToTalkButton/>
      </PushToTalkButtonContainer>
    </div>
    <div>
      <DetailsTracktitle="Income"/>
      <DetailsTracktitle="Expense"/>
    </div>
  </div>

);
};
export default Dashboard;

```

## Dashboard.js

```

import {makeStyles} from '@material-ui/core/styles';

export default makeStyles((theme) => ({
  desktop: {
    [theme.breakpoints.up('sm')]: {display:
      'none',

```

```

    },
  },
  mobile:[{theme.breakpoints.down('
    sm')]:{
      display:'none',
    },
  },
  last:{
    [theme.breakpoints.down('sm')]:{
      marginBottom:theme.spacing(
        3),paddingBottom:'200px',
    },
  },
  grid:{
    '>*':{
      margin:theme.spacing(2),
    },
  },
});

```

## Main.jsx

```

import React from 'react';
import {Card,CardContent,Typography,Grid,Divider} from '@mui/material';im
port './main.css';
import Form from './Form/Form';i
mport List from './List/List';impor
t {useContext} from 'react';
import {ExpenseTrackerContext} from '../context/context
';import InfoCard from '../InfoCard';

```

```
import useTransactions from '../..useTransactions';
```

```
const Main = () => {
```

```
  const { balance } = useContext(ExpenseTrackerContext);
```

```
  const { total: incomeTotal } = useTransactions("Income");
```

```
  const { total: expenseTotal } = useTransactions("Expense");
```

```
  return (
```

```
    <Card style={{ width: '370px', padding: '10px', backgroundColor: 'transparent' }}>
```

```
      /*<CardHeader className="cardHeader" title="Personal Expense Tracker" subheader="speak to track"/>*/
```

```
      <CardContent className="cardContent1">
```

```
        <Typography style={{ color: 'rgb(144, 122, 0)' }} align="center" variant="h6">Total
```

```
        balance: <span style={{ color: 'goldenrod' }}>Rs. {balance}</span>
```

```
      </Typography>
```

```
        <Typography style={{ color: 'green' }} align="center" variant="h6">Income: <
```

```
        span style={{ color: 'rgba(0, 255, 0, 0.9)' }}>Rs. {incomeTotal}</span>
```

```
      </Typography>
```

```
        <Typography style={{ color: 'red' }} align="center" variant="h6">Expense:
```

```
        <span style={{ color: 'rgba(255, 0, 0, 0.8)' }}>Rs. {expenseTotal}</span>
```

```
      </Typography>
```

```
    <Typography variant="subtitle1" style={{ lineHeight: '1.5em', marginTop: '20px' }}>
```

```
      <InfoCard />
```

```
    </Typography>
```

```

        <Dividerstyle={{borderTop:'1pxdashedgray',backgroundColor:'white',paddingTop:
'15px'}}/>
        <Form/>
      </CardContent>
      <CardContentclassName="cardContent2">
        <Gridcontainerspacing={2}>
          <Griditemxs={12}>
            <List/>
          </Grid>
        </Grid>
      </CardContent>
    </Card>
  );
};

```

```
exportdefaultMain;
```

## main.css

```

.cardHeader{
  text-align:center;
  border:2pxsolidrgb(1441220);
  color:rgb(1441220);bor
  der-
  radius:5px;background-
  color:white;
}

.cardContent1{
  border:2pxsolidrgb(1441220);
  border-radius:5px;
  background-color:white;
}

```

```
.cardContent2{
  border:2pxsolidrgb(1441220);
  border-radius:5px;
  background-color:white;
}
```

## Form.jsx

```
import React,{useContext,useEffect,useState}from'react';
import{TextField,Typography,Grid,FormControl,InputLabel,Select,MenuItem,OutlinedInput}
from '@mui/material';
import{v4asuuidv4}from'uuid';import'./
form.css';
import{ExpenseTrackerContext}from'../../context/context';
import{expenseCategories,incomeCategories}from'../../constants/categories';i
mportformatDatefrom'../../utils/formatDate';
import{useSpeechContext}from'@speechly/react-
client';importCustomizedSnackbarfrom'../../Snackbar/S
nackbar';
```

```
constinitialState={amoun
  t:"",
  category:"",typ
  e:'Income',
  date:formatDate(newDate()),
}
```

```
constForm=()=>=>{

  const[formData,setFormData]=useState(initialState);
  const{addTransaction}=useContext(ExpenseTrackerContext)
  ;const{segment}=useSpeechContext();
```

```

const[open,setOpen]=useState(false);
const[unfill,setUnfill]=useState(false);

constcreateTransaction=()=>{

  if(formData.category&&formData.amount){

    consttransaction={...formData,amount:Number(formData.amount),id:uuidv4()}

    addTransaction(transaction);se
    tFormData(initialState);setOpen
    (true);
  }
  else{
    setUnfill(true);
    /*swal("UnfulfilledInputs","Pleasefillallthefields","warning");*/
  }
}

useEffect(()=>{

  if(segment){
    if(segment.intent.intent==='add_expense'){se
      tFormData({...formData,type:'Expense'});
    }
    elseif(segment.intent.intent==='add_income'){setFormData
      ({...formData,type:'Income'});
    }
    elseif(segment.isFinal&&segment.intent.intent==='create_transaction'){retu
      rncreateTransaction();

```

```

    }

    elseif(segment.isFinal&&segment.intent.intent==='cancel_transaction'){return
      rnssetData(initialState);
    }

    segment.entities.forEach((e)=>{
      constcategory=`${e.value.charAt(0)}${e.value.slice(1).toLowerCase()}`;

      switch(e.type){case'amount':
        setFormData({...formData,amount:e.value});
        break;
        case'category':
          if(incomeCategories.map((ic)=>ic.type).includes(category)){setForm
            mData({...formData,type:'Income',category});
          }
          elseif(expenseCategories.map((ec)=>ec.type).includes(category)){setForm
            Data({...formData,type:'Expense',category});
          }
          break;case'
        se'date':
          setFormData({...formData,date:e.value});
          break;
          default:
            break;
        }
      });

      if(segment.isFinal&&formData.type&&formData.category&&formData.amount&
        &formData.date){
        createTransaction();
      }
    }
  }
}

```



```
}  
}
```

```
//eslint-disable-next-line  
,[segment]);
```

```
const selectedCategories = formData.type === 'Income' ? incomeCategories : expenseCategories;
```

```
return(  
  <Grid container spacing={2}>  
    <CustomizedSnackbar title="form" open={open} setOpen={setOpen}/>  
    <CustomizedSnackbar title="unfilled" open={unfill} setOpen={setUnfill}/>  
    <Grid item xs={12}>  
      <Typography align="center" variant="subtitle2" gutterBottom>  
        {segment &&  
          segment.words.map((w) => w.value).join("")  
        }  
      </Typography>  
    </Grid>  
    <Grid item xs={6}>  
      <FormControl fullWidth>  
        <InputLabel>Type</InputLabel>  
        <Select  
          value={formData.type}  
          onChange={(e) => setFormData({...formData, type: e.target.value})} input={<OutlinedInput label="Type"/>  
        >  
          <MenuItem value="Income">Income</MenuItem>  
          <MenuItem value="Expense">Expense</MenuItem>  
        </Select>
```

```

        </FormControl>
    </Grid>
    <Griditemxs={6}>
        <FormControlfullWidth>
            <InputLabel>Category</InputLabel>
            <Select
                value={formData.category}
                onChange={(e)=>setFormData({...formData,category:e.target.value})}input
                =<OutlinedInputlabel="Category"/>
            >
                {
                    selectedCategories.map((c)=>
                        <MenuItemkey={c.type}value={c.type}>
                            {c.type}
                        </MenuItem>)
                }
            </Select>
        </FormControl>
    </Grid>
    <Griditemxs={6}>
        <TextFieldtype="number"onKeyDown={(e)=>{e.key==='e'&&e.preventDefault();
e.key==='-'&&e.preventDefault();
e.key==='+'&&e.preventDefault();}}label="Amount"fullWidthvalue={formData.amount}o
nChange={(e)
=>setFormData({...formData,amount:e.target.value})}/>
    </Grid>
    <Griditemxs={6}>
        <TextFieldtype="date"label="Date"fullWidthvalue={formData.date}onChan
ge={(e)=>setFormData({...formData,date:formatDate(e.target.value)})}/>
    </Grid>
    <buttonclassName="createBtn"onClick={createTransaction}>Create</button>
</Grid>
);

```

```
};
```

```
export default Form;
```

## form.css

```
.createBtn{  
  margin-top:50px;margin-bottom:20px;margin-left:16px;width:100%;  
  border-radius:10px;padding:15px;  
  border:2px solid rgb(111 950);  
  background-color:rgb(231 229 175);cursor:pointer;  
  color:rgb(111 950);  
  font-weight:600;  
}
```

## List.jsx

```
import React from 'react';  
  
import { List as MUIList, ListItem, ListItemAvatar, ListItemText, Avatar, ListItemSecondaryAction, IconButton, Slide } from '@mui/material';  
  
import CurrencyRupeeIcon from '@mui/icons-material/CurrencyRupee';  
import DeleteIcon from '@mui/icons-material/Delete';  
  
import { useContext } from 'react';
```

```
import{ExpenseTrackerContext}from'../../context/context';import{useState}from'react';
```

```

import CustomizedSnackBar
from'../Snackbar/Snackbar';import'./list.css';

const List=()=>{

  const {deleteTransaction,transactions}=useContext(ExpenseTrackerContext);c
  onst [open,setOpen]=useState(false);

  return(
    <MUIListdense={false}className='listContainer'>
      <CustomizedSnackBartitle="list"open={open)setOpen={setOpen}/>

      {transactions.map((transaction)=>(
        <Slidedirection="down"inmountOnEnterunmountOnExitkey={transaction.id}>
          <ListItem>
            <ListItemAvatar>
              <Avatarstyle={{backgroundColor:transaction.type==='Income'?'green':
'red'}}>

                <CurrencyRupeeIcon/>
              </Avatar>
            </ListItemAvatar>
            <ListItemTextprimary={transaction.category}
secondary={`Rs.${transaction.amount}-${transaction.date}`}/>
            <ListItemSecondaryAction>
              <IconButtonedge="end"aria-
label="delete"onClick={()=>{setOpen(true);deleteTransaction(transaction.id)}}>
                <DeleteIcon/>
              </IconButton>
            </ListItemSecondaryAction>
          </ListItem>

```

```

        </Slide>
      )))
    </MuiList>
  );
}

```

```
export default List;
```

## ListStyles.js

```

import {makeStyles} from '@material-
ui/core/styles';import {red,green} from '@material-
ui/core/colors';

```

```

export default makeStyles((theme)=>({
  avatarIncome:{
    color:
      '#fff',backgroundColor:gree
      n[500],
  },
  avatarExpense:{
    color:theme.palette.getContrastText(red[500]),backg
    roundColor:red[500],
  },
  list:{
    maxHeight:'240px',o
    verflow:'auto',
  },
})));

```

## list.css

```
.avatarIncome{
```

```
    color:#fff;
    background-color:green;
  }
```

```
.avatarExpense{
  color:#fff;
  background-color:red;
}
```

```
.listContainer{
  max-
  height:240px;overflow:
  auto;
}
```

## **context.js**

```
import React,{useReducer,createContext}from'react';
```

```
import contextReducer from './contextReducer';
```

```
const initialState=JSON.parse(localStorage.getItem('transaction'))||[];
```

```
export const ExpenseTrackerContext=createContext(initialState);
```

```
export const Provider=({children})=>{
```

```
  const [transactions,dispatch]=useReducer(contextReducer,initialState);
```

```
  const deleteTransaction=(id)=>dispatch({type:'DELETE_TRANSACTION',payload:id});
```

```
  const addTransaction= (transaction)=>
  dispatch({ type:'ADD_TRANSACTION',payload:transaction});
```

```
const balance= transactions.reduce((acc,currVal) => (currVal.type=== 'Expense' ? acc - currVal.amount:acc+currVal.amount),0);
```

```
return(
  <ExpenseTrackerContext.Providervalue={{deleteTransaction,
    addTransaction,
    transactions,balance,
  }}>
    {children}
  </ExpenseTrackerContext.Provider>
);
};
```

## InfoCard.jsx

```
import React from 'react';
```

```
const isIncome=Math.round(Math.random());
```

```
const InfoCard=()=>{return(
  <div style={{textAlign: 'center', padding: '0 10%', paddingBottom:'5%'}}>Try saying:<br />
    Add{isIncome?'Income':'Expense'}for{isIncome?'Rs.100':'Rs.50'}
    inCategory{isIncome?'Business':'House'}for{isIncome?'Monday':'Friday'}
  </div>
```



```
);  
};
```

```
export default InfoCard;
```

## **useTransaction.jsx**

```
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;
    }
}
elseif(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;bre
        ak;case'12':monthExpenseTotal[11].amount+=t.amount;b
        reak;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={data
  sets:[
    {
      label:'Expense',
      data:monthExpenseTotal.map((m)=>m.amount),borderC
      olor:'#b50d12',
      backgroundColor:'#e57c58',t
      ension:0.1,
    },
  ],
  labels:monthExpenseTotal.map((m)=>m.m)
};

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

export default useTransactions;

```