**Frontend**

**Desktop\6thsem\ExpenseTracker\ET\client\public\index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<link

href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"

crossorigin="anonymous"

/>

<title>Expense Managment System</title>

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>

**Desktop\6thsem\ExpenseTracker\ET\client\src\components\Layout\Footer.js**

import React from "react";

const Footer = () => {

return (

<div className="bg-dark text-light p-4">

<h6 className="text-center">All rights reserved &copy; ET</h6>

</div>

);

};

export default Footer;

**Desktop\6thsem\ExpenseTracker\ET\client\src\components\Layout\Header.js**

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

import { Link, useNavigate } from "react-router-dom";

import { message } from "antd";

const Header = () => {

const [loginUser, setLoginUser] = useState("");

const navigate = useNavigate();

useEffect(() => {

const user = JSON.parse(localStorage.getItem("user"));

if (user) {

setLoginUser(user);

}

}, []);

const logoutHandler = () => {

localStorage.removeItem("user");

message.success("Logout Successfully");

navigate("/login");

};

return (

<nav className="navbar navbar-expand-lg bg-light">

<div className="container-fluid">

<button

className="navbar-toggler"

type="button"

data-bs-toggle="collapse"

data-bs-target="#navbarTogglerDemo01"

aria-controls="navbarTogglerDemo01"

aria-expanded="false"

aria-label="Toggle navigation"

>

<span className="navbar-toggler-icon" />

</button>

<div className="collapse navbar-collapse" id="navbarTogglerDemo01">

<Link className="navbar-brand" to="/">

Expense Tracker

</Link>

<ul className="navbar-nav ms-auto mb-2 mb-lg-0">

<li className="nav-item">

{loginUser && <p className="nav-link">{loginUser.name}</p>}

</li>

<li className="nav-item">

<button className="logout-button" onClick={logoutHandler}>

Logout

</button>

</li>

</ul>

</div>

</div>

</nav>

);

};

export default Header;

**Desktop\6thsem\ExpenseTracker\ET\client\src\components\Layout\Layout.js**

import React from "react";

import Footer from "./Footer";

import Header from "./Header";

const Layout = ({ children }) => {

return (

<>

<Header />

<div className="content">{children}</div>

<Footer />

</>

);

};

export default Layout;

**Desktop\6thsem\ExpenseTracker\ET\client\src\components\Analytics.js**

import React from "react";

import { Progress } from "antd";

const Analytics = ({ allTransaction }) => {

// category

const categories = [

"salary",

"tip",

"bill",

"food",

"tax",

"project",

"medicine",

"fees",

];

// totaltransaction

const totalTransaction = allTransaction.length;

const totalIncomeTransactions = allTransaction.filter(

(transaction) => transaction.type === "income"

);

const totalExpenseTransactions = allTransaction.filter(

(transaction) => transaction.type === "expense"

);

const totalIncomePercent =

(totalIncomeTransactions.length / totalTransaction) \* 100;

const totalExpensePercent =

(totalExpenseTransactions.length / totalTransaction) \* 100;

// totalturnover

const totalTurnover = allTransaction.reduce(

(acc, transaction) => acc + transaction.amount,

0

);

const totalIncomeTurnover = allTransaction

.filter((transaction) => transaction.type === "income")

.reduce((acc, transaction) => acc + transaction.amount, 0);

const totalExpenseTurnover = allTransaction

.filter((transaction) => transaction.type === "expense")

.reduce((acc, transaction) => acc + transaction.amount, 0);

const totalIncomeTurnoverPercent =

(totalIncomeTurnover / totalTurnover) \* 100;

const totalExpenseTurnoverPercent =

(totalExpenseTurnover / totalTurnover) \* 100;

return (

<>

<div className="row m-3">

<div className="col-md-4">

<div className="card">

<div className="card-header">

total transactions:{totalTransaction}

</div>

<div className="card-body">

<h5 className="text-success">

Income:{totalIncomeTransactions.length}

</h5>

<h5 className="text-danger">

Expense:{totalExpenseTransactions.length}

</h5>

<div>

<Progress

type="circle"

strokeColor={"green"}

className="mx-2"

percent={totalIncomePercent.toFixed(0)}

/>

<Progress

type="circle"

strokeColor={"red"}

className="mx-2"

percent={totalExpensePercent.toFixed(0)}

/>

</div>

</div>

</div>

</div>

<div className="col-md-4">

<div className="card">

<div className="card-header">total Turnover:{totalTurnover}</div>

<div className="card-body">

<h5 className="text-success">Income:{totalIncomeTurnover}</h5>

<h5 className="text-danger">Expense:{totalExpenseTurnover}</h5>

<div>

<Progress

type="circle"

strokeColor={"green"}

className="mx-2"

percent={totalIncomeTurnoverPercent.toFixed(0)}

/>

<Progress

type="circle"

strokeColor={"red"}

className="mx-2"

percent={totalExpenseTurnoverPercent.toFixed(0)}

/>

</div>

</div>

</div>

</div>

</div>

<div className="row mt-3">

<div className="col-md-5">

<h4>Categorywise income</h4>

{categories.map((category) => {

const amount = allTransaction

.filter(

(transaction) =>

transaction.type === "income" &&

transaction.category === category

)

.reduce((acc, transaction) => acc + transaction.amount,0);

return(

amount>0&&(

<div className="card">

<div className="card-body">

<h5>{category}</h5>

<Progress percent={((amount/totalIncomeTurnover)\*100).toFixed(0)}/>

</div>

</div>

)

);

})}

</div>

<div className="col-md-5">

<h4>Categorywise expense</h4>

{categories.map((category) => {

const amount = allTransaction

.filter(

(transaction) =>

transaction.type === "expense" &&

transaction.category === category

)

.reduce((acc, transaction) => acc + transaction.amount,0);

return(

amount>0&&(

<div className="card">

<div className="card-body">

<h5>{category}</h5>

<Progress percent={((amount/totalExpenseTurnover)\*100).toFixed(0)}/>

</div>

</div>

)

);

})}

</div>

</div>

</>

);

};

export default Analytics;

**Desktop\6thsem\ExpenseTracker\ET\client\src\components\Spinner.js**

import React from "react";

const Spinner = () => {

return (

<>

<div className="d-flex justify-content-center">

<div className="spinner-border" role="status">

<span className="visually-hidden">Loading...</span>

</div>

</div>

</>

);

};

export default Spinner;

**Desktop\6thsem\ExpenseTracker\ET\client\src\pages\HomePage.js**

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

import { Modal, Form, Input, Select, message, Table, DatePicker } from "antd";

import {

UnorderedListOutlined,

AreaChartOutlined,

EditOutlined,

DeleteOutlined,

} from "@ant-design/icons";

import Layout from "./../components/Layout/Layout";

import axios from "axios";

import Spinner from "../components/Spinner";

import moment from "moment";

import "../index.css";

import Analytics from "../components/Analytics";

const { RangePicker } = DatePicker;

const HomePage = () => {

const [showModal, setShowModal] = useState(false);

const [loading, setLoading] = useState(false);

const [allTransaction, setAllTransaction] = useState([]);

const [frequency, setFrequency] = useState("7");

const [selectedDate, setSelectedate] = useState([]);

const [type, setType] = useState("all");

const [viewData, setViewData] = useState("table");

const [editable,setEditable]=useState(null)

// table data

const columns = [

{

title: "Date",

dataIndex: "date",

render: (text) => <span>{moment(text).format("YYYY-MM-DD")}</span>,

},

{

title: "Amount",

dataIndex: "amount",

},

{

title: "Type",

dataIndex: "type",

},

{

title: "Category",

dataIndex: "category",

},

{

title: "Reference",

dataIndex: "reference",

},

{

title: "Actions",

render:(text,record)=>(

<div>

<EditOutlined onClick={()=>{

setEditable(record)

setShowModal(true)

}}/>

<DeleteOutlined className="mx-2" onClick={()=>{handleDelete(record)}}/>

</div>

)

},

];

// useEffect Hook

useEffect(() => {

// Get all transactions

const getAllTransaction = async () => {

try {

const user = JSON.parse(localStorage.getItem("user"));

setLoading(true);

const res = await axios.post("/transections/get-transection", {

userid: user.\_id,

frequency,

selectedDate,

type,

});

setLoading(false);

setAllTransaction(res.data);

console.log(res.data);

} catch (error) {

console.log(error);

message.error(" fetch Issue with transactions");

}

};

getAllTransaction();

}, [frequency, selectedDate, type]);

// delete handler

const handleDelete=async(record)=>{

try {

setLoading(true)

await axios.post("/transections/delete-transection",{transectionId:record.\_id})

setLoading(false)

message.success("transaction deleted");

} catch (error) {

setLoading(false)

console.log(error);

message.error('unable to delete');

}

};

// Form handling

const handleSubmit = async (values) => {

try {

const user = JSON.parse(localStorage.getItem("user"));

setLoading(true);

// await axios.post("/transections/add-transection", {

// ...values,

// userid: user.\_id,

// type: values.type,

// });

if(editable){

await axios.post("/transections/edit-transection",{

payload:{

...values,

userId:user.\_id,

},

transectionId:editable.\_id

});

setLoading(false);

message.success("Transaction updated successfully");

}else{

await axios.post("/transections/add-transection",{

...values,

userid:user.\_id,

});

}

setLoading(false);

message.success("Transaction added successfully");

setShowModal(false);

setEditable(null);

} catch (error) {

setLoading(false);

message.error("Failed to add transaction");

}

};

return (

<Layout>

{loading && <Spinner />}

<div className="filters">

<div>

<h6>Select Frequency</h6>

<Select value={frequency} onChange={(values) => setFrequency(values)}>

<Select.Option value="7">LAST 1 Week</Select.Option>

<Select.Option value="31">LAST 1 Month</Select.Option>

<Select.Option values="365">LAST 1 Year</Select.Option>

<Select.Option value="custom">Custom</Select.Option>

</Select>

{frequency === "custom" && (

<RangePicker

value={selectedDate}

onChange={(values) => setSelectedate(values)}

/>

)}

</div>

<div>

<h6>Select Type</h6>

<Select value={type} onChange={(values) => setType(values)}>

<Select.Option value="all">All</Select.Option>

<Select.Option value="income">Income</Select.Option>

<Select.Option values="expense">Expense</Select.Option>

</Select>

{frequency === "custom" && (

<RangePicker

value={selectedDate}

onChange={(values) => setSelectedate(values)}

/>

)}

</div>

<div className="switch-icons">

<UnorderedListOutlined

className={`mx-2 ${

viewData === "table" ? "active-icon" : "inactive-icon"

}`}

onClick={() => setViewData("table")}

/>

<AreaChartOutlined

className={`mx-2 ${

viewData === "analytics" ? "active-icon" : "inactive-icon"

}`}

onClick={() => setViewData("analytics")}

/>

</div>

<div>

<button

className="btn btn-primary"

onClick={() => setShowModal(true)}

>

Add New

</button>

</div>

</div>

<div className="content">

{viewData === "table" ? (

<Table columns={columns} dataSource={allTransaction} />

) : (

<Analytics allTransaction={allTransaction} />

)}

</div>

<Modal

title={editable?"edit transaction":"add transaction"}

open={showModal}

onCancel={() => setShowModal(false)}

footer={null}

>

<Form layout="vertical" onFinish={handleSubmit}initialValues={editable}>

<Form.Item label="Amount" name="amount">

<Input type="text" />

</Form.Item>

<Form.Item

label="Type"

name="type"

rules={[{ required: true, message: "Please select a type" }]}

>

<Select>

<Select.Option value="income">Income</Select.Option>

<Select.Option value="expense">Expense</Select.Option>

</Select>

</Form.Item>

<Form.Item label="Category" name="category">

<Select>

<Select.Option value="salary">Salary</Select.Option>

<Select.Option value="tip">Tip</Select.Option>

<Select.Option value="bill">Bill</Select.Option>

<Select.Option value="food">Food</Select.Option>

<Select.Option value="tax">Tax</Select.Option>

<Select.Option value="project">Project</Select.Option>

<Select.Option value="medicine">Medicine</Select.Option>

<Select.Option value="fees">Fees</Select.Option>

</Select>

</Form.Item>

<Form.Item label="Date" name="date">

<Input type="date" />

</Form.Item>

<Form.Item label="Reference" name="reference">

<Input type="text" />

</Form.Item>

<Form.Item label="Description" name="description">

<Input type="text" />

</Form.Item>

<div className="d-flex justify-content-end">

<button type="submit" className="btn btn-primary">

SAVE

</button>

</div>

</Form>

</Modal>

</Layout>

);

};

export default HomePage;

**Desktop\6thsem\ExpenseTracker\ET\client\src\pages\Login.js**

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

import { Form, Input, message } from "antd";

import { Link, useNavigate } from "react-router-dom";

import axios from "axios";

import Spinner from "../components/Spinner";

const Login = () => {

const [loading, setLoading] = useState(false);

const navigate = useNavigate();

// Form submit

const submitHandler = async (values) => {

try {

setLoading(true);

const { data } = await axios.post("/users/login", values);

setLoading(false);

message.success("Login success");

localStorage.setItem("user", JSON.stringify({ ...data.user, password: "" }));

navigate("/");

} catch (error) {

setLoading(false);

message.error("Something went wrong");

}

};

// Prevent for logged-in user

useEffect(() => {

if (localStorage.getItem("user")) {

navigate("/");

}

}, [navigate]);

return (

<div className="login-page">

{loading && <Spinner />}

<Form layout="vertical" onFinish={submitHandler} className="login-form">

<h1>Login Form</h1>

<Form.Item

label="Email"

name="email"

rules={[

{ required: true, message: "Please enter your email" },

{ type: "email", message: "Please enter a valid email" },

]}

>

<Input type="email" />

</Form.Item>

<Form.Item

label="Password"

name="password"

rules={[{ required: true, message: "Please enter your password" }]}

>

<Input type="password" />

</Form.Item>

<div className="d-flex">

<div className="register-link">

<Link to="/register">Not a user? Click Here to Register</Link>

</div>

<div className="login-button">

<button className="btn btn-primary">Login</button>

</div>

</div>

</Form>

</div>

);

};

export default Login;

**Desktop\6thsem\ExpenseTracker\ET\client\src\pages\Register.js**

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

import { Form, Input, message } from "antd";

import { Link, useNavigate } from "react-router-dom";

import axios from "axios";

import Spinner from "../components/Spinner";

const Register = () => {

const navigate = useNavigate();

const [loading, setLoading] = useState(false);

// Form submit

const submitHandler = async (values) => {

try {

setLoading(true);

await axios.post("/users/register", values);

message.success("Registration Successful");

setLoading(false);

navigate("/login");

} catch (error) {

setLoading(false);

message.error("Something went wrong");

}

};

// Prevent for logged-in user

useEffect(() => {

if (localStorage.getItem("user")) {

navigate("/");

}

}, [navigate]);

return (

<div className="register-page">

{loading && <Spinner />}

<Form

layout="vertical"

onFinish={submitHandler}

className="register-form"

>

<h1>Register Form</h1>

<Form.Item

label="Name"

name="name"

rules={[

{ required: true, message: "Please enter your name" },

{ min: 3, message: "Name must be at least 3 characters long" },

]}

>

<Input />

</Form.Item>

<Form.Item

label="Email"

name="email"

rules={[

{ required: true, message: "Please enter your email" },

{ type: "email", message: "Please enter a valid email" },

]}

>

<Input type="email" />

</Form.Item>

<Form.Item

label="Password"

name="password"

rules={[

{ required: true, message: "Please enter your password" },

{ min: 6, message: "Password must be at least 6 characters long" },

]}

>

<Input type="password" />

</Form.Item>

<div className="register-buttons">

<div className="register-link">

<Link to="/login">Already Registered? Click Here to Login</Link>

</div>

<div className="register-button">

<button className="btn btn-primary">Register</button>

</div>

</div>

</Form>

</div>

);

};

export default Register;

**Desktop\6thsem\ExpenseTracker\ET\client\src\App.js**

import { Routes, Route, Navigate } from "react-router-dom";

import HomePage from "./pages/HomePage";

import Login from "./pages/Login";

import Register from "./pages/Register";

function App() {

return (

<>

<Routes>

<Route

path="/"

element={

<ProtectedRoutes>

<HomePage />

</ProtectedRoutes>

}

/>

<Route path="/register" element={<Register />} />

<Route path="/login" element={<Login />} />

</Routes>

</>

);

}

export function ProtectedRoutes(props) {

if (localStorage.getItem("user")) {

return props.children;

} else {

return <Navigate to="/login" />;

}

}

export default App;

**Desktop\6thsem\ExpenseTracker\ET\client\src\index.css**

\* {

padding: 0;

margin: 0;

box-sizing: border-box;

}

.content {

height: 120vh;

}

.register-page {

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

height: 100vh;

background-image: url("https://tse4.mm.bing.net/th?id=OIP.NfFhTSxEndiClX3FbMakxQHaE8&pid=Api&P=0&h=180");

background-size: cover;

}

.register-form {

width: 300px;

}

.register-form .ant-form-item {

margin-bottom: 10px;

}

.register-buttons {

display: flex;

justify-content: space-between;

align-items: center;

margin-top: 10px;

padding: 10px;

border: 1px solid #ccc;

border-radius: 5px;

}

.register-buttons .register-link {

flex: 1;

margin-right: 10px;

}

.register-buttons .register-button {

margin-left: 10px;

}

/\* .resgister-page {

display: flex;

align-items: center;

justify-content: center;

height: 100vh;

} \*/

/\* CSS for Login page \*/

.login-page {

position: relative;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

height: 100vh;

background-image: url("https://tse2.mm.bing.net/th?id=OIP.FLeRI6metHwujMau6\_MAEwHaEK&pid=Api&P=0&h=180");

background-size: cover;

}

.login-page::before {

content: "";

position: absolute;

top: -10px;

right: -10px;

bottom: -10px;

left: -10px;

border: 10px ;

border-radius: 10px;

z-index: -1;

}

.login-form {

width: 300px;

padding: 20px;

border: 1px solid #ccc;

border-radius: 5px;

box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);

}

.login-form .ant-form-item {

margin-bottom: 10px;

}

.login-form .d-flex {

display: flex;

justify-content: space-between;

align-items: center;

margin-top: 10px;

}

.login-form .d-flex .register-link {

flex: 1;

text-align: left;

}

.login-form .d-flex .login-button {

margin-left: 10px;

}

/\* CSS for Header \*/

.navbar {

background: linear-gradient(to right, #ff6b6b, #0c00ff);

color: #f4eded;

padding: 10px 20px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

font-family: 'Courier New', Courier, monospace;

}

.navbar-brand {

font-size: 20px;

font-weight: bold;

color: #000;

text-decoration: none;

}

.navbar-nav .nav-item {

margin-left: 10px;

}

.nav-link {

color: #000;

text-decoration: none;

}

.logout-button {

padding: 6px 10px;

background-color: #007bff;

color: #fff;

border: none;

border-radius: 5px;

cursor: pointer;

}

.logout-button:hover {

background-color: #0056b3;

}

.navbar-toggler {

border: none;

padding: 0;

background-color: transparent;

}

.navbar-toggler .navbar-toggler-icon {

background-color: #000;

width: 20px;

height: 2px;

}

/\* transaction page\*/

.filters{

display: flex;

align-items: center;

justify-content: space-between;

padding: 15px 20px;

box-shadow: 0 0 3px gray;

}

/\*atnd icons\*/

.switch-icons {

border: 1px solid black rgba(0, 0, 0, 0.677);

border-radius: 5px;

padding: 10px 15px;

}

.anticon svg {

font-size: 20px;

cursor: pointer;

}

.active-icon {

color: black;

}

.inactive-icon {

color: gray;

}

**Desktop\6thsem\ExpenseTracker\ET\client\src\index.js**

import React from "react";

import ReactDOM from "react-dom/client";

import "antd/dist/reset.css";

import "./index.css";

import App from "./App";

import { BrowserRouter } from "react-router-dom";

import reportWebVitals from "./reportWebVitals";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(

<React.StrictMode>

<BrowserRouter>

<App />

</BrowserRouter>

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**Desktop\6thsem\ExpenseTracker\ET\client\package.json**

{

"proxy": "http://localhost:8080/api/v1",

"name": "client",

"version": "0.1.0",

"private": true,

"dependencies": {

"@ant-design/icons": "^5.1.2",

"@testing-library/jest-dom": "^5.14.1",

"@testing-library/react": "^13.0.0",

"@testing-library/user-event": "^13.2.1",

"antd": "^5.5.1",

"axios": "^1.4.0",

"moment": "^2.29.4",

"react": "^18.2.0",

"react-dom": "^18.2.0",

"react-redux": "^8.0.5",

"react-router": "^6.11.2",

"react-router-dom": "^6.11.2",

"react-scripts": "5.0.1",

"redux": "^4.2.1",

"web-vitals": "^2.1.0"

},

"scripts": {

"start": "react-scripts start",

"build": "react-scripts build",

"test": "react-scripts test",

"eject": "react-scripts eject"

},

"eslintConfig": {

"extends": [

"react-app",

"react-app/jest"

]

},

"browserslist": {

"production": [

">0.2%",

"not dead",

"not op\_mini all"

],

"development": [

"last 1 chrome version",

"last 1 firefox version",

"last 1 safari version"

]

}

}

**Backend**

**Desktop\6thsem\ExpenseTracker\ET\config\connectDb.js**

const mongoose = require("mongoose");

const colors = require("colors");

const connectDb = async () => {

try {

await mongoose.connect(process.env.MONGO\_URL);

console.log(`Server Running On ${mongoose.connection.host}`.bgCyan.white);

} catch (error) {

console.log(`${error}`.bgRed);

}

};

module.exports = connectDb;

**Desktop\6thsem\ExpenseTracker\ET\controllers\transectionCtrl.js**

const transectionModel = require("../models/transectionModel");

const moment = require('moment');

const getAllTransection = async (req, res) => {

try {

const { frequency, selectedDate, type } = req.body;

// const { userid } = req.body; // Extract the userid from req.body

const transections = await transectionModel.find({

// userid: userid, // Use the extracted userid

...(frequency !== "custom"

? {

date: {

$gt: moment().subtract(Number(frequency), "d").toDate(),

},

}

: {

date: {

$gte: selectedDate[0],

$lte: selectedDate[1],

},

}),

userid: req.body.userid,

...(type !== "all" && { type }),

});

res.status(200).json(transections);

} catch (error) {

console.log(error);

res.status(500).json(error);

}

};

const deleteTransection =async(req,res)=>{

try {

await transectionModel.findByIdAndDelete({

\_id:req.body.transectionId});

res.status(200).send('transaction deleted')

} catch (error) {

console.log(erroe);

res.status(500).json(error)

}

};

const editTransection=async(req,res)=>{

try {

await transectionModel.findOneAndUpdate({\_id:req.body.transectionId},req.body.payload);

res.status(200).send('Edit Successfully')

} catch (error) {

console.log(error);

res.status(500).json(error)

}

};

const addTransection = async (req, res) => {

try {

const newTransection = new transectionModel(req.body);

await newTransection.save();

res.status(201).send("Transection Created");

} catch (error) {

console.log(error);

res.status(500).json(error);

}

};

module.exports = { getAllTransection, addTransection,editTransection,deleteTransection };

**Desktop\6thsem\ExpenseTracker\ET\controllers\userController.js**

const userModel = require("../models/userModel");

// login callback

const loginController = async (req, res) => {

try {

const { email, password } = req.body;

const user = await userModel.findOne({ email, password });

if (!user) {

return res.status(404).send("User Not Found");

}

res.status(200).json({

success: true,

user,

});

} catch (error) {

res.status(400).json({

success: false,

error,

});

}

};

//Register Callback

const registerController = async (req, res) => {

try {

const newUser = new userModel(req.body);

await newUser.save();

res.status(201).json({

success: true,

newUser,

});

} catch (error) {

res.status(400).json({

success: false,

error,

});

}

};

module.exports = { loginController, registerController };

**Desktop\6thsem\ExpenseTracker\ET\models\transectionModel.js**

const mongoose =require('mongoose')

const transectionSchema=new mongoose.Schema({

userid: {

type: String,

required: true,

},

amount: {

type: Number,

required: [true, "amount is required"],

},

type: {

type: String,

required: [true, "type is required"],

},

category: {

type: String,

requires: [true, "cat is required"],

},

reference: {

type: String,

},

description: {

type: String,

required: [true, "desc is required"],

},

date: {

type: Date,

required: [true, "data is required"],

},

},

{ timestamps: true }

);

const transectionModel = mongoose.model('transections',transectionSchema)

module.exports = transectionModel;

**Desktop\6thsem\ExpenseTracker\ET\models\userModel.js**

const mongoose = require("mongoose");

//schema design

const userSchema = new mongoose.Schema(

{

name: {

type: String,

required: [true, "name is required"],

},

email: {

type: String,

required: [true, "email is required and should be unique"],

unique: true,

},

password: {

type: String,

required: [true, "password is required"],

},

},

{ timestamps: true }

);

//export

const userModel = mongoose.model("users", userSchema);

module.exports = userModel;

**Desktop\6thsem\ExpenseTracker\ET\routes\transectionRoutes.js**

const express =require('express');

const {addTransection, getAllTransection,editTransection,deleteTransection } = require('../controllers/transectionCtrl');

//router object

const router=express.Router();

//routes

//add transection POST MEthod

router.post("/add-transection", addTransection);

//edit transection POST MEthod

router.post("/edit-transection", editTransection);

//add transection POST MEthod

router.post("/delete-transection", deleteTransection);

//get transactions

router.post('/get-transection', getAllTransection);

module.exports=router;

**Desktop\6thsem\ExpenseTracker\ET\routes\userRoute.js**

const express = require("express");

const {

loginController,

registerController,

} = require("../controllers/userController");

//router object

const router = express.Router();

//routers

// POST || LOGIN USER

router.post("/login", loginController);

//POST || REGISTER USER

router.post("/register", registerController);

module.exports = router;

**Desktop\6thsem\ExpenseTracker\ET\.env**

PORT=8080

MONGO\_URL=mongodb+srv://nishmi:nishmi@cluster.nvmxlgt.mongodb.net/expanseApp

**Desktop\6thsem\ExpenseTracker\ET\server.js**

const express = require("express");

const cors = require("cors");

const morgan = require("morgan");

const dotenv = require("dotenv");

const colors = require("colors");

const connectDb = require("./config/connectDb");

// config dot env file

dotenv.config();

//databse call

connectDb();

//rest object

const app = express();

//middlewares

app.use(morgan("dev"));

app.use(express.json());

app.use(cors());

//routes

app.use("/api/v1/users", require("./routes/userRoute"));

//transections routes

app.use("/api/v1/transections", require("./routes/transectionRoutes"));

//port

const PORT = 8080 || process.env.PORT;

//listen server

app.listen(PORT, () => {

console.log(`Server running on port ${PORT}`);

});

**Desktop\6thsem\ExpenseTracker\ET\package.json**

{

"name": "et",

"version": "1.0.0",

"description": "expense tracker",

"main": "server.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1",

"start": "node server.js",

"server": "nodemon server.js",

"client": "yarn start --prefix client",

"dev": "concurrently \"npm run server\"\"yarn run client\""

},

"keywords": [],

"author": "",

"license": "MIT",

"dependencies": {

"colors": "^1.4.0",

"concurrently": "^8.0.1",

"cors": "^2.8.5",

"dotenv": "^16.0.3",

"express": "^4.18.2",

"moment": "^2.29.4",

"mongoose": "^7.2.0",

"morgan": "^1.10.0",

"nodemon": "^2.0.22"

}

}