B1: cài vnpay npm

B2: Server

import { VNPay } from "vn-payments";

/\* eslint-disable no-param-reassign \*/

const TEST\_CONFIG = VNPay.TEST\_CONFIG;

const vnpay = new VNPay({

  paymentGateway: TEST\_CONFIG.paymentGateway,

  merchant: TEST\_CONFIG.merchant,

  secureSecret: TEST\_CONFIG.secureSecret,

});

orderRouter.post("/payment/checkout", (req, res) => {

  const userAgent = req.headers["user-agent"];

  const params = Object.assign({}, req.body);

  const { order } = params;

  const clientIp =

    req.headers["x-forwarded-for"] ||

    req.connection.remoteAddress ||

    req.socket.remoteAddress ||

    (req.connection.socket ? req.connection.socket.remoteAddress : null);

  const amount = parseInt(order.totalPrice);

  const now = new Date();

  const checkoutData = {

    amount,

    clientIp: clientIp.length > 15 ? "127.0.0.1" : clientIp,

    locale: "vn",

    billingCity: order.shippingAddress.city || "",

    billingPostCode: order.shippingAddress.postalCode || "",

    billingStateProvince: params.billingStateProvince || "",

    billingStreet: params.billingStreet || "",

    billingCountry: order.shippingAddress.country || "",

    deliveryAddress: order.shippingAddress.address || "",

    deliveryCity: order.shippingAddress.city || "",

    deliveryCountry: order.shippingAddress.country || "",

    currency: "VND",

    deliveryProvince: params.billingStateProvince || "",

    customerEmail: order.user.email,

    customerPhone: params.phoneNumber || "",

    orderId: `node-${now.toISOString()}`,

    // returnUrl: ,

    transactionId: `node-${now.toISOString()}`,

    customerId: order.user.email,

    returnUrl: `http://${req.headers.host}/api/orders/payment/vnpay/callback`,

    orderInfo: "Shope",

    orderType: "Medicine",

  };

  vnpay.buildCheckoutUrl(checkoutData).then((checkoutUrl) => {

    if (checkoutUrl) {

      res.json({ url: checkoutUrl.href });

    } else {

      res.send("Payment method not found");

    }

  });

});

// tra ve tu vnpal

orderRouter.get(

  "/payment/:gateway/callback",

  asyncHandler(async (req, res) => {

    const vnpResponseCode = req.query.vnp\_ResponseCode;

    if (vnpResponseCode === "00") {

      console.log("thanh cong");

      // update status order

    } else {

      console.log("that bai");

    }

  })

);

B3: client

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

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

import Header from "./../components/Header";

import { PayPalButton } from "react-paypal-button-v2";

import { useDispatch, useSelector } from "react-redux";

import { getOrderDetails, payOrder } from "../Redux/Actions/OrderAction";

import Loading from "../components/LoadingError/Loading";

import Message from "../components/LoadingError/Error";

import moment from "moment";

import axios from "axios";

import { ORDER\_PAY\_RESET } from "../Redux/Constants/OrderConstant";

import { OnePayInternational, VNPay } from "vn-payments";

import urlParse from "url-parse";

/\* eslint-disable no-param-reassign \*/

const OrderScreen = () => {

  window.scrollTo(0, 0);

  const { loading, error, order } = useSelector((state) => state.orderDetails);

  const { id: idParam } = useParams();

  const dispatch = useDispatch();

  const [paymentUrl, setPaymentUrl] = useState("");

  const [sdkReady, setSdkReady] = useState(true);

  const {

    loading: loadingPay,

    error: errorPay,

    success,

  } = useSelector((state) => state.orderPay);

  if (!loading && !error) {

    order.itemsPrice = order.orderItems

      .reduce((sum, current) => sum + current.price \* current.qty, 0)

      .toFixed(2);

  }

  useEffect(() => {

    // chưa có thông tin order hoặc success là true (cập nhật lại orderDetail)

    if (!order || success) {

      dispatch({ type: ORDER\_PAY\_RESET });

      dispatch(getOrderDetails(idParam));

    }

    // chưa trả

    else if (!order.isPaid) {

      setSdkReady(true);

    }

    // trả rồi

    else if (order.isPaid) {

      setSdkReady(false);

    }

  }, [dispatch, idParam, success, order]);

  const handlePayment = async () => {

    const response = await axios.post("/api/orders/payment/checkout", {

      order,

    });

    setPaymentUrl(response.data.url);

  };

  if (paymentUrl) {

    window.location.href = paymentUrl;

 }

 console.log(paymentUrl)

  return (

    <>

      <Header />

      <div className="container">

        {loading ? (

          <Loading />

        ) : error ? (

          <Message>{error}</Message>

        ) : order ? (

          <>

            <div className="row  order-detail">

              <div className="col-lg-4 col-sm-4 mb-lg-4 mb-5 mb-sm-0">

                <div className="row">

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

                    <div className="alert-success order-box">

                      <i className="fas fa-user"></i>

                    </div>

                  </div>

                  <div className="col-md-8 center">

                    <h5>

                      <strong>Customer</strong>

                    </h5>

                    <p>{order.user.name}</p>

                    <p>

                      <a href={`mailto:admin@example.com`}>

                        {order.user.email}

                      </a>

                    </p>

                  </div>

                </div>

              </div>

              {/\* 2 \*/}

              <div className="col-lg-4 col-sm-4 mb-lg-4 mb-5 mb-sm-0">

                <div className="row">

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

                    <div className="alert-success order-box">

                      <i className="fas fa-truck-moving"></i>

                    </div>

                  </div>

                  <div className="col-md-8 center">

                    <h5>

                      <strong>Order info</strong>

                    </h5>

                    <p>Shipping: {order.shippingAddress.country}</p>

                    <p>Pay method: {order.paymentMethod}</p>

                    {order.isPaid ? (

                      <div className="bg-info p-2 col-12">

                        <p className="text-white text-center text-sm-start">

                          Paid on

                          {moment(order.createdAt).calendar()}

                        </p>

                      </div>

                    ) : (

                      <div className="bg-danger p-2 col-12">

                        <p className="text-white text-center text-sm-start">

                          Not paid

                        </p>

                      </div>

                    )}

                  </div>

                </div>

              </div>

              {/\* 3 \*/}

              <div className="col-lg-4 col-sm-4 mb-lg-4 mb-5 mb-sm-0">

                <div className="row">

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

                    <div className="alert-success order-box">

                      <i className="fas fa-map-marker-alt"></i>

                    </div>

                  </div>

                  <div className="col-md-8 center">

                    <h5>

                      <strong>Deliver to</strong>

                    </h5>

                    <p>Address: {order.shippingAddress.address}</p>

                    {order.isDelivered ? (

                      <div className="bg-danger p-1 col-12">

                        <p className="text-white text-center text-sm-start">

                          Delivered

                        </p>

                      </div>

                    ) : (

                      <div className="bg-danger p-1 col-12">

                        <p className="text-white text-center text-sm-start">

                          Not Delivered

                        </p>

                      </div>

                    )}

                  </div>

                </div>

              </div>

            </div>

            <div className="row order-products justify-content-between">

              <div className="col-lg-8">

                {order.orderItems.map((order, index) => {

                  return (

                    <div key={index} className="order-product row">

                      <div className="col-md-3 col-6">

                        <img src={order.image} alt="product" />

                      </div>

                      <div className="col-md-5 col-6 d-flex align-items-center">

                        <Link to={`/`}>

                          <h6>{order.name}</h6>

                        </Link>

                      </div>

                      <div className="mt-3 mt-md-0 col-6 col-md-2  d-flex align-items-center flex-column justify-content-center ">

                        <h4>QUANTITY</h4>

                        <h6>{order.qty}</h6>

                      </div>

                      <div className="mt-3 mt-md-0 col-md-2 col-6 align-items-end  d-flex flex-column justify-content-center">

                        <h4>SUBTOTAL</h4>

                        <h6>${order.price \* order.qty}</h6>

                      </div>

                    </div>

                  );

                })}

              </div>

              <div className="col-lg-3 d-flex align-items-end flex-column mt-5 subtotal-order">

                <table className="table table-bordered">

                  <tbody>

                    <tr>

                      <td>

                        <strong>Products</strong>

                      </td>

                      <td>${order.itemsPrice}</td>

                    </tr>

                    <tr>

                      <td>

                        <strong>Shipping</strong>

                      </td>

                      <td>${order.shippingPrice}</td>

                    </tr>

                    <tr>

                      <td>

                        <strong>Tax</strong>

                      </td>

                      <td>${order.taxPrice}</td>

                    </tr>

                    <tr>

                      <td>

                        <strong>Total</strong>

                      </td>

                      <td>${order.totalPrice}</td>

                    </tr>

                  </tbody>

                </table>

                <div className="col-12">

                  <button onClick={handlePayment}>Thanh toán</button>

                </div>

              </div>

            </div>

          </>

        ) : (

          ""

        )}

      </div>

    </>

  );

};

export default OrderScreen;