✅ Thay calculateOrder tĩnh bằng dữ liệu thật từ CartContext.

**✅ MỤC TIÊU:**

CardDetails.jsx sẽ tính tổng đơn hàng từ giỏ hàng thật (cartItems) + shipping.

**🪜 BƯỚC 1: Viết lại file orderUtils.js**

👉 **Bạn mở orderUtils.js và thay TOÀN BỘ bằng nội dung sau:**

js

Sao chépChỉnh sửa

// orderUtils.js

// ✅ Hàm cũ nếu vẫn dùng ở nơi khác

export function calculateOrder({ quantity, shippingMethod }) {

const price = 1000;

const shipping = shippingMethod === "express" ? 20 : 10;

const subtotal = quantity \* price;

const total = subtotal + shipping;

return { subtotal, shipping, total };

}

// ✅ Hàm mới dùng với dữ liệu từ giỏ hàng

export function calculateOrderFromCart(cartItems = [], shippingMethod = "standard") {

const subtotal = cartItems.reduce((sum, item) => {

return sum + item.price \* item.quantity;

}, 0);

const shipping = shippingMethod === "express" ? 20 : 10;

const total = subtotal + shipping;

return { subtotal, shipping, total };

}

✅ Xong bước 1 → Ghi lại nhé: ✔️ orderUtils.js đã ok!

**🪜 BƯỚC 2: Sửa FormikProps.jsx**

👉 Bạn mở FormikProps.jsx và làm 2 việc:

**📌 a. Xoá dòng quantity khỏi initialValues:**

js

Sao chépChỉnh sửa

const initialValues = {

cardholderName: "",

cardNumber: "",

exp: "",

cvv: "",

shippingMethod: "standard",

cardType: "",

};

**📌 b. Xoá quantity khỏi validationSchema:**

js

Sao chépChỉnh sửa

const validationSchema = Yup.object({

cardholderName: Yup.string()

.matches(/^[A-Za-z\s'-]+$/, "Tên không hợp lệ")

.required("Bắt buộc"),

cardNumber: Yup.string()

.matches(/^[0-9]{16}$/, "Số thẻ phải gồm 16 chữ số")

.required("Bắt buộc"),

exp: Yup.string()

.required("Bắt buộc")

.matches(/^(0[1-9]|1[0-2])\/\d{4}$/, "Định dạng không hợp lệ (MM/YYYY)"),

cvv: Yup.string()

.required("Bắt buộc")

.matches(/^\d{3,4}$/, "CVV phải là 3 hoặc 4 số"),

shippingMethod: Yup.string()

.required("Chọn phương thức giao hàng")

.oneOf(["standard", "express"], "Không hợp lệ"),

});

**📌 c. Xoá onSubmit ở file này luôn (để chuyển về CardDetails.jsx xử lý):**

js

Sao chépChỉnh sửa

// ❌ Xoá đoạn này đi

// const onSubmit = (values) => {

// ...

// }

export const formikProps = {

initialValues,

validationSchema,

};

✅ Xong bước 2 → Ghi lại: ✔️ FormikProps.jsx đã làm sạch!

**🪜 BƯỚC 3: Sửa CardDetails.jsx để kết nối giỏ hàng**

👉 Mở CardDetails.jsx, sửa như sau:

**📌 a. Import các phần cần:**

js

Sao chépChỉnh sửa

import { useCart } from "../context/CartContext"; // lấy cartItems

import { calculateOrderFromCart } from "./orderUtils"; // hàm mới

import { formikProps } from "./FormikProps"; // lấy initialValues + validationSchema

**📌 b. Thêm useCart() trong component:**

js

Sao chépChỉnh sửa

const { cartItems } = useCart();

**📌 c. Trong <FormikWrapper>, sửa onSubmit:**

jsx

Sao chépChỉnh sửa

onSubmit={(values) => {

const { total } = calculateOrderFromCart(cartItems, values.shippingMethod);

console.log("Tổng:", total);

console.log("Giá trị form:", values);

}}

**📌 d. Thay calculateOrder(values) bằng:**

js

Sao chépChỉnh sửa

const { subtotal, shipping, total } = calculateOrderFromCart(

cartItems,

values.shippingMethod

);

✅ Xong bước 3 → Ghi lại: ✔️ CardDetails.jsx đã kết nối context + tính tổng

**🪜 BƯỚC 4: Xoá input quantity (vì lấy từ giỏ hàng)**

Bạn xoá đoạn này khỏi form:

jsx

Sao chépChỉnh sửa

{/\* Quantity \*/}

<div className="mb-3">

...

</div>

Thay vào đó, nếu muốn **hiển thị sản phẩm trong giỏ hàng**, bạn có thể thêm:

jsx

Sao chépChỉnh sửa

<div className="mb-4">

<p className="text-white fw-bold">Order Summary</p>

<ul className="list-group">

{cartItems.map((item) => (

<li key={item.id} className="list-group-item d-flex justify-content-between bg-transparent text-white border-white">

<span>{item.name} x {item.quantity}</span>

<span>${(item.price \* item.quantity).toFixed(2)}</span>

</li>

))}

</ul>

</div>

✅ Xong bước 4 → Ghi lại: ✔️ Đã bỏ quantity tĩnh & hiển thị giỏ hàng động

**🎉 XONG!**

Khi bạn làm xong cả 4 bước, form của bạn sẽ:

* Dùng Formik cho các trường nhập thẻ.
* Lấy sản phẩm từ giỏ hàng thật (cartItems).
* Tính subtotal, shipping, total theo giỏ hàng và shipping method.
* Gọn gàng, đúng luồng, dễ bảo trì.