XÂY DỰNG ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ

XÂY DỰNG THANH TOÁN VNPAY

1. **Mục đích & yêu cầu**

   a. **Mục đích:**

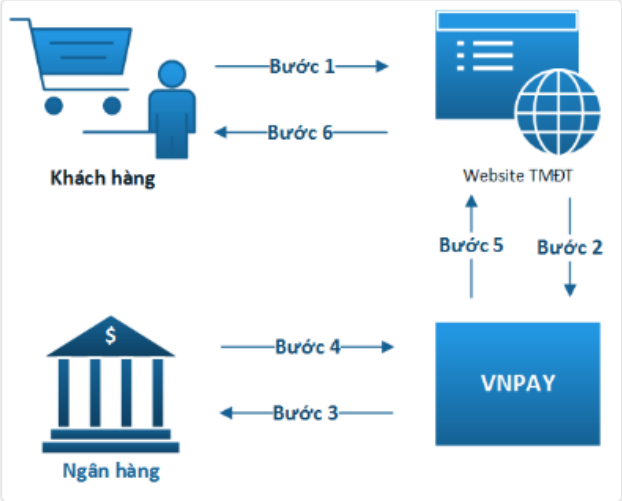
- Giới thiệu về khái niệm thanh toán và phương thức thanh toán trong thương mại điện tử
- Hiểu và triển khai được phương thức thanh toán trên ứng dụng thương mại điện tử của mình.

   b. **Yêu cầu:**

- Sinh viên ***đọc kỹ bài hướng dẫn*** và thực hiện các bước để nắm rõ quy trình xây dựng phương thức thanh toán.

2. **Tổng quan về xây dựng phương thức thanh toán VNPAY**

- **Mô hình kết nối của ví điện tử VNPAY**



Bước 1: Khách hàng thực hiện mua hàng trên Website - ứng dụng TMĐT và tiến hành thanh toán trực tuyến cho đơn hàng.

Bước 2: Website - ứng dụng TMĐT thành l**ập yêu cầu thanh toán dưới dạng URL** mang thông tin thanh toán và c**huyển hướng khách hàng** sang Cổng thanh toán VNPAY bằng URL đó.

Cổng thanh toán VNPAY xử lý yêu cầu thanh toán mà Website - ứng dụng TMĐT gửi sang. Khách hàng tiến hành **nhập hoặc xử lý xác thực các thông tin được yêu cầu Thanh toán.**

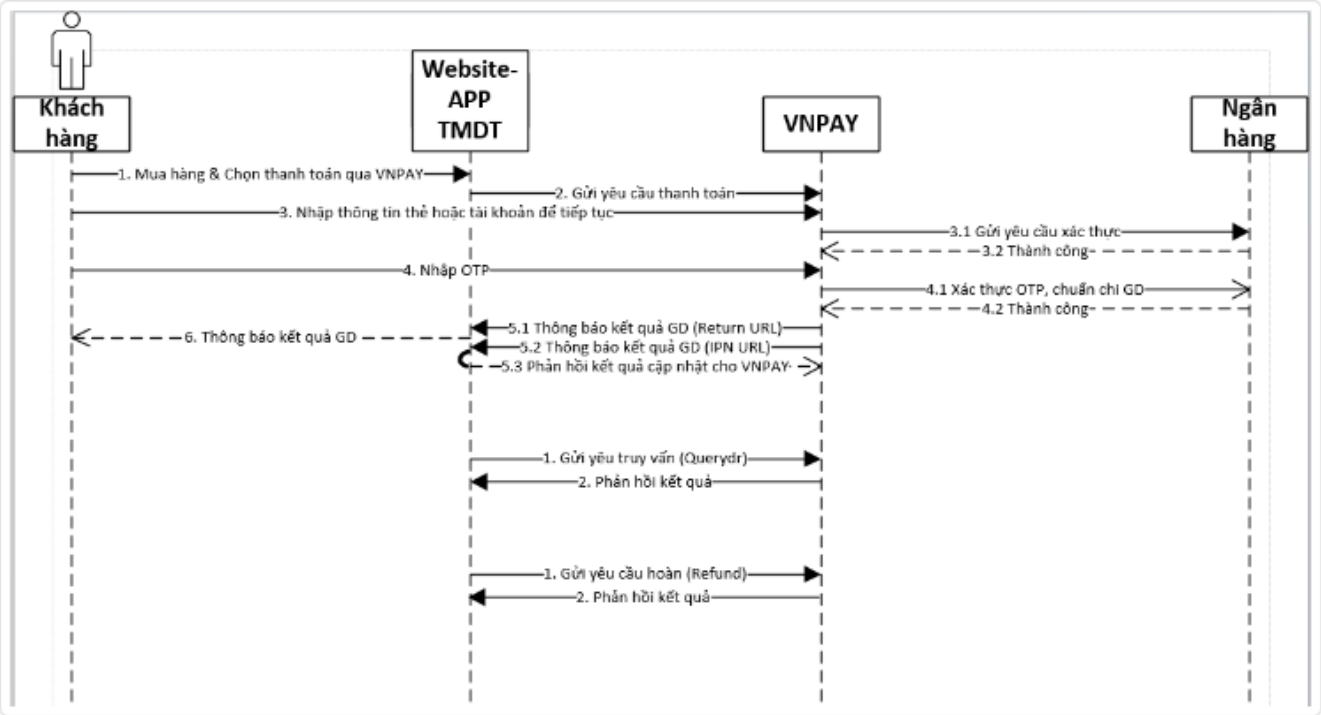
Bước 3,4: Khách hàng nhập thông tin để xác minh tài khoản Ngân hàng của khách hàng và xác thực giao dịch (Nhập thông tin tài khoản, thẻ hoặc quét mã VNPAY-QR).

Bước 5: Giao dịch thành công tại Ngân hàng, VNPAY tiến hành:

Chuyển hướng khách hàng về Website - ứng dụng TMĐT. Thông báo cho Website - ứng dụng TMĐT kết quả thanh toán của khách hàng thông qua IPN URL. Merchant cập nhật kết quả thanh toán VNPAY gửi tại URL này.

Bước 6: Merchant hiển thị kết quả giao dịch tới khách hàng.

Sơ đồ tuần tự:

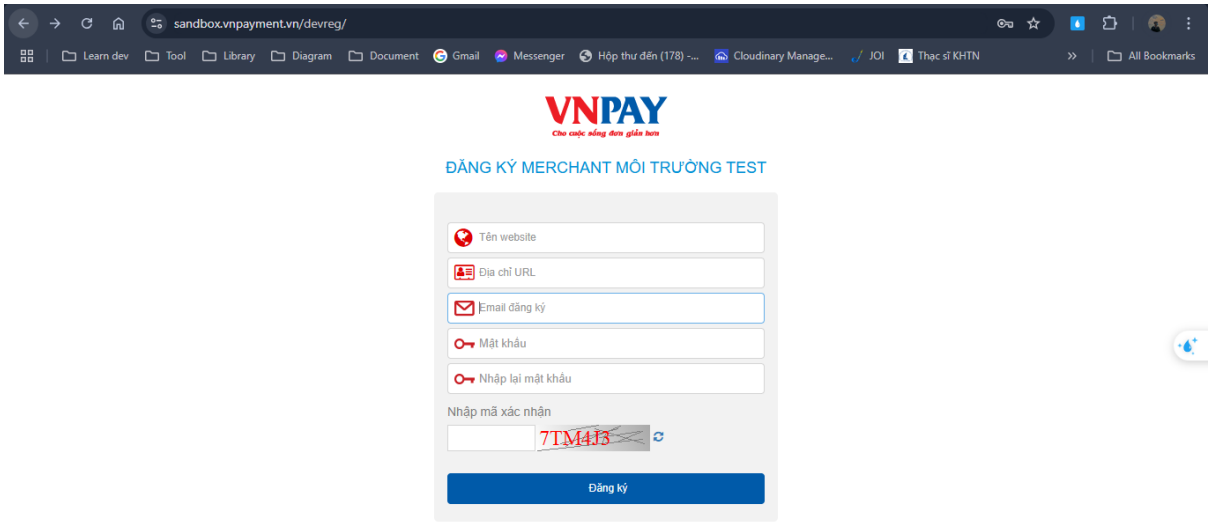


3. Các bước thực hiện cấu hình thanh toán VNPAY vào hệ thống.

Phần 1: Đăng ký VNPAY Merchant

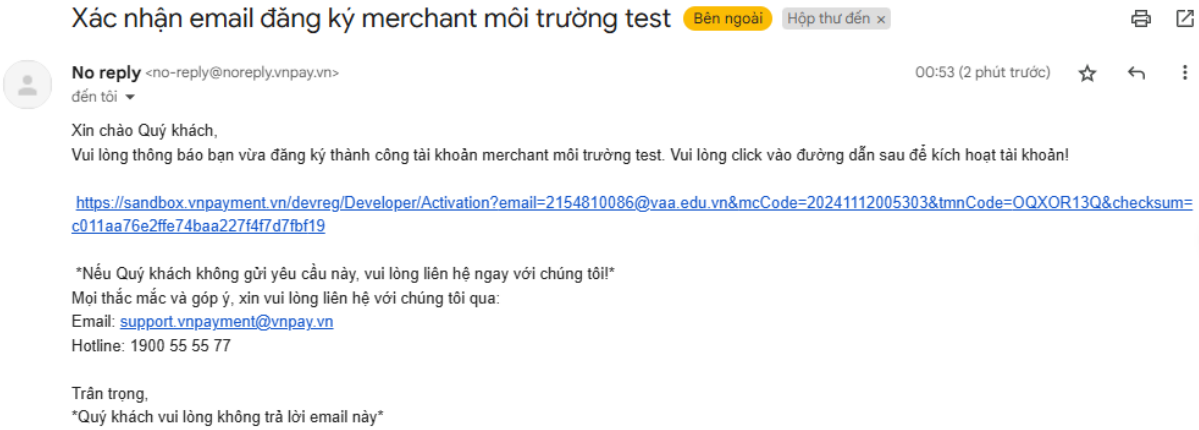
Thực hiện đăng ký dịch vụ VNPAY test merchant

https://sandbox.vnpayment.vn/devreg/



Đăng ký thành công và check mail

Kích hoạt tài khoản và làm theo hướng dẫn



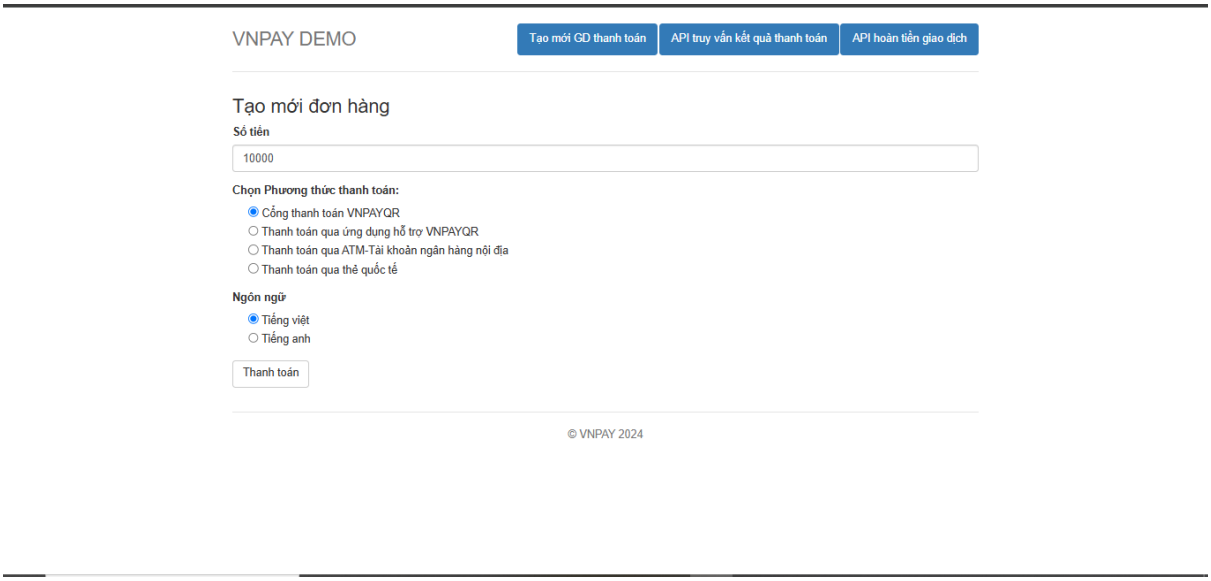
VNPAY gửi thông tin hướng dẫn cấu hình và tích hợp hệ thống.

Tải code tích hợp của VNPAY của node.js.

https://drive.google.com/file/d/1AUTr9CKF4W7_VNqDZCmSTf7ElNOcgaf9/view?usp=sharing

Phần 2: Cấu hình gửi yêu cầu thanh toán từ frontend

Khi run dự án của vnpay demo ta sẽ thấy có giao diện như sau:



Như vậy, mỗi khi gửi dữ liệu để tạo một thanh toán mới, ta cần gửi đi các thông tin như sau:

- **Tổng số tiền đơn hàng**
- **Phương thức thanh toán**
- **Ngôn ngữ hiển thị**
- **Thông tin đơn hàng: sử dụng để tạo đơn hàng.**

Như vậy ở phía frontend ta sẽ gửi một API Post tạo URL thanh toán với các dữ liệu đi kèm trong request body.

Tạo một button thanh toán với VNPAY và xử lý sự kiện onClick của nút thanh toán như sau:

```
async function handlePayment() {
  try {
    const total = totalMoney(selectedProducts);
    const newPayment = {
      products: selectedProducts,
      amount: total,
      bankCode: null,
      language: "vn",
    };
    const response = await axios.post(
      `http://localhost:8080/api/v1/vnpay/create_payment_url`,
      newPayment
    );
    if (response.status === 200 && response.data) {
      window.location.href = response.data;
    }
  } catch (error) {
    alert(`Lỗi: ${error?.message}`)
  }

}
```

Trường hợp gửi **bankCode** = null, hệ thống sẽ hiểu rằng cho người dùng chọn phương thức thanh toán mà VNPAY hỗ trợ.

Phần 3: Cấu hình tích hợp VNPAY vào backend của hệ thống

Khi người dùng click vào nút thanh toán, website TMDT sẽ gửi các thông tin về đơn hàng để server tạo đơn hàng và tạo URL thanh toán, sau đó sẽ chuyển hướng người dùng đến VNPAY.

Các bước cần xử lý tích hợp code cài đặt

(1) Cài đặt code build URL thanh toán chuyển hướng và tạo đơn hàng.

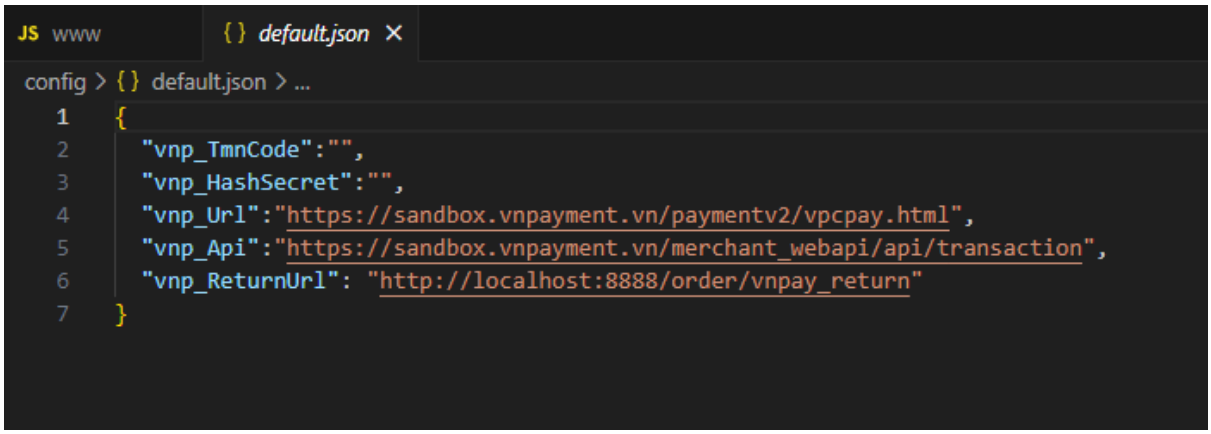
(2) Cài đặt code **vnp_ReturnUrl** URL thông báo kết quả thanh toán và cập nhật trạng thái thanh toán đơn hàng.

(3) Cài đặt code IPN URL cập nhật kết quả thanh toán. Gửi lại VNPAY URL này khi thiết lập xong.

Hướng dẫn tích hợp:

Tại dự án **backend của bạn**, tạo **folder config** và tạo **file default.json,** thay thế thông tin **vnp_TmnCode** và **vnp_HashSecret** bằng thông tin được gửi đến email của bạn, chỉnh sửa port của **vnp_ReturnUrl** thành url và port của localserver.

Tạo url thanh toán và đơn hàng, chuyển hướng



Tạo 1 router URL dành riêng cho thanh toán VNPAY

```
const vnpay = require('./routers/vnpay')
app.use(`${api}/vnpay`, vnpay)
```

Cài đặt các thư viện cần thiết cho cấu hình thanh toán VNPAY:

npm install config md5 moment qs set-tz sha256

(trong trường hợp thiếu thư viện nào, sinh viên tiến hành cài đặt thư viện bổ sung)

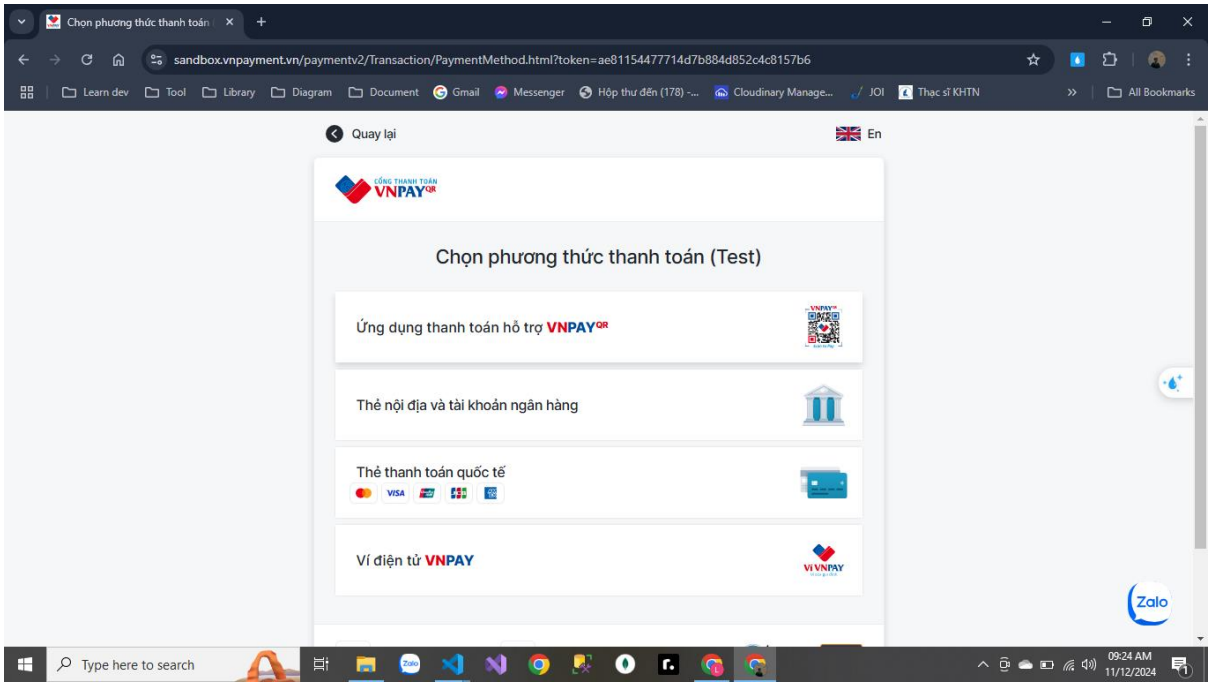
Tạo file **vnpay.js** và tích hợp các route thanh toán như **create_payment_url, vnpay_return, vnpay_ipn**

Trong trường hợp website của sinh viên đã triển khai các API đơn hàng, tiến hành tạo đơn hàng khi tạo URL

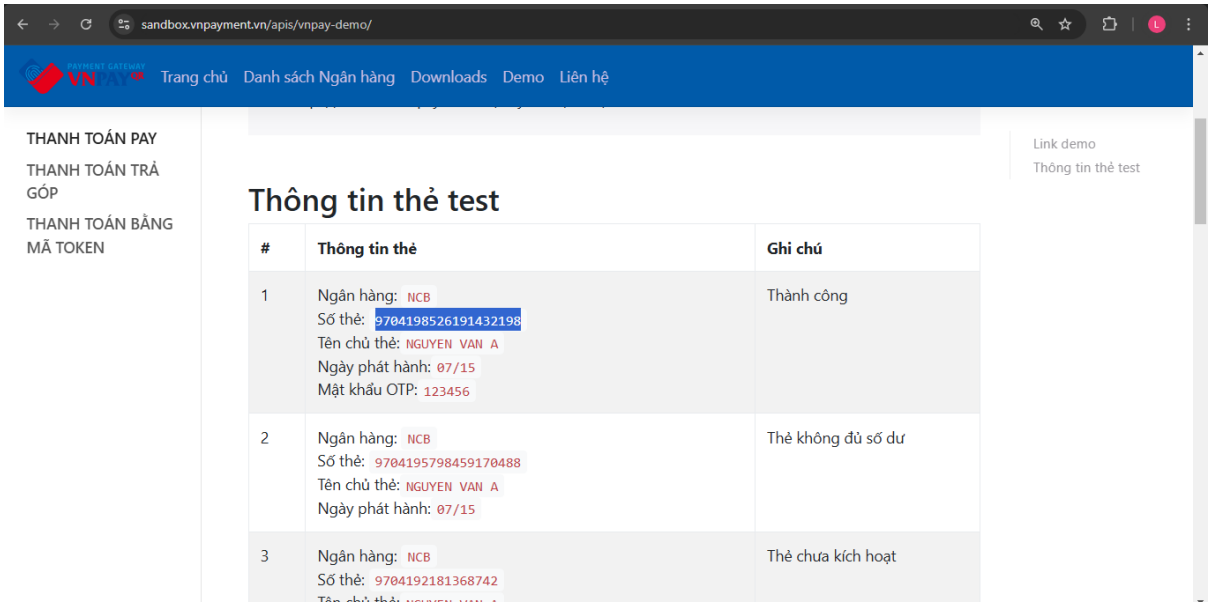
```
router.post('/create_payment_url', function (req, res, next) {
    process.env.TZ = 'Asia/Ho_Chi_Minh'
    let date = new Date()
    let createDate = moment(date).format('YYYYMMDDHHmmss')
    let ipAddr =
        req.headers['x-forwarded-for'] ||
        req.connection.remoteAddress ||
        req.socket.remoteAddress ||
        req.connection.socket.remoteAddress
    let config = require('config')
    let tmnCode = config.get('vnp_TmnCode')
    let secretKey = config.get('vnp_HashSecret')
    let vnpUrl = config.get('vnp_Url')
```

```
    let returnUrl = config.get('vnp_ReturnUrl')
    let orderId = moment(date).format('DDHHmmss')
    let amount = req.body.amount
    let bankCode = req.body.bankCode
    let locale = req.body.language
    if (locale === null || locale === '') {
        locale = 'vn'
    }
    let currCode = 'VND'
    let vnp_Params = {}
    vnp_Params['vnp_Version'] = '2.1.0'
    vnp_Params['vnp_Command'] = 'pay'
    vnp_Params['vnp_TmnCode'] = tmnCode
    vnp_Params['vnp_Locale'] = locale
    vnp_Params['vnp_CurrCode'] = currCode
    vnp_Params['vnp_TxnRef'] = orderId
    vnp_Params['vnp_OrderInfo'] = 'Thanh toan cho ma GD:' + orderId
    vnp_Params['vnp_OrderType'] = 'other'
    vnp_Params['vnp_Amount'] = amount * 100
    vnp_Params['vnp_ReturnUrl'] = returnUrl
    vnp_Params['vnp_IpAddr'] = ipAddr
    vnp_Params['vnp_CreateDate'] = createDate
    if (bankCode !== null && bankCode !== '') {
        vnp_Params['vnp_BankCode'] = bankCode
    }

    vnp_Params = sortObject(vnp_Params)

    let querystring = require('qs')
    let signData = querystring.stringify(vnp_Params, { encode: false })
    let crypto = require('crypto')
    let hmac = crypto.createHmac('sha512', secretKey)
    let     signed     =     hmac.update(new     Buffer(signData,     'utf-
8')).digest('hex')
    vnp_Params['vnp_SecureHash'] = signed
    vnpUrl += '?' + querystring.stringify(vnp_Params, { encode: false })

    // GỬI VỀ URL ĐỂ FRONTEND CHUYỂN HƯỚNG ĐẾN VNPAY
    res.send(vnpUrl)
})
```

Sau khi nhận được URL thanh toán, frontend sẽ chuyển hướng người dùng đến VNPAY

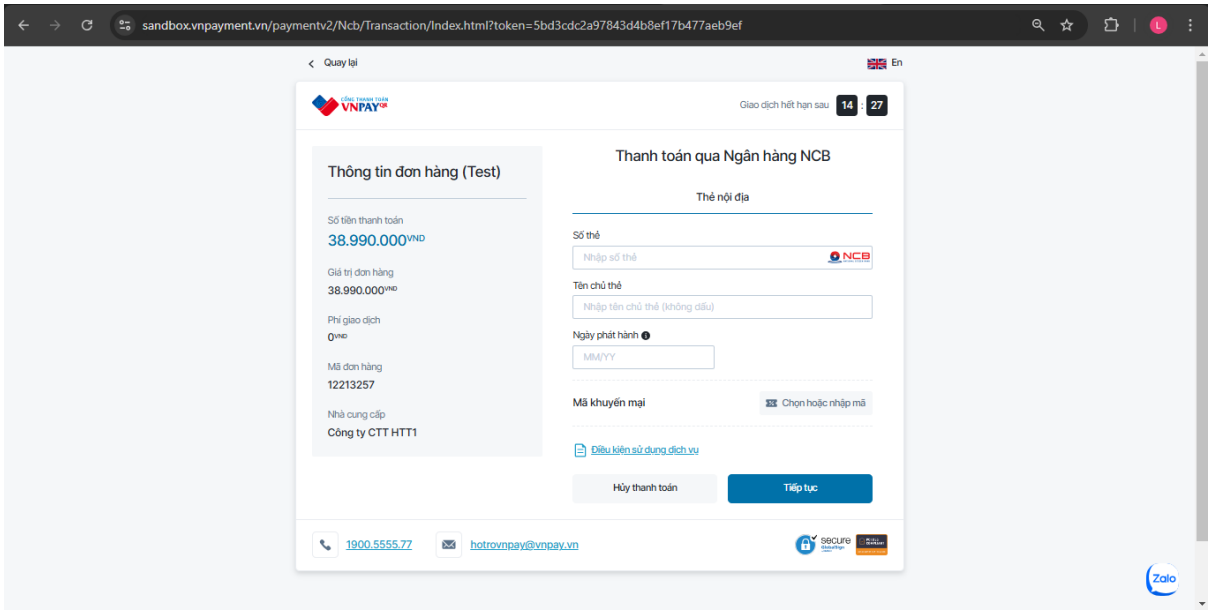


Trên môi trường test, VNPAY cung cấp cho chúng ta tài khoản test như sau:



Chúng ta chọn hình thức thanh toán và tài khoản thanh toán, điền thông tin và tiếp tục thanh toán.

Nhập thông tin thanh toán



Nhập mã pin thanh toán

Khi thanh toán thành công sẽ chuyển hướng người dùng về lại trang thanh toán thành công.

Xử lý thanh toán và thông báo đến người dùng

```
router.get('/vnpay_return', function (req, res, next) {
    let vnp_Params = req.query;

    let secureHash = vnp_Params['vnp_SecureHash'];

    delete vnp_Params['vnp_SecureHash'];
    delete vnp_Params['vnp_SecureHashType'];

    vnp_Params = sortObject(vnp_Params);

    let config = require('config');
    let tmnCode = config.get('vnp_TmnCode');
    let secretKey = config.get('vnp_HashSecret');

    let querystring = require('qs');
    let signData = querystring.stringify(vnp_Params, { encode: false });
    let crypto = require("crypto");
    let hmac = crypto.createHmac("sha512", secretKey);
    let signed = hmac.update(new Buffer(signData, 'utf-
8')).digest("hex");
```

```
    if(secureHash === signed){
        //Kiem tra xem du lieu trong db co hop le hay khong va thong bao
ket qua

        const redirectToPaymentSuccessPage = (status) => `
            <html>
                <head>
                    <meta http-equiv="refresh" content="0;
url=http://localhost:3000/">
                </head>
                <body>
                    <p>Redirecting to payment ${status} page...</p>
                </body>
            </html>`

        if (orderStatus === 'success') {
            res.status(200).send(redirectToPaymentSuccessPage('success'))
        } else {
            res.status(200).send(redirectToPaymentSuccessPage('error'))
        }
});
```