Hướng dẫn xây dựng Website   
Thương mại Điện tử với ASP.NET Core MVC 8

ASP.NET Core MVC 8 - E-Commerce Project

*HCM, 29/11/2023*

## **MỤC LỤC**

[**MỤC LỤC 1**](#_l6co5x6codbk)

[**1. Mục đích 1**](#_guy751zh8rsw)

[**2. Yêu cầu 1**](#_n03pswx37z4l)

[**3. Nội dung thực hiện 1**](#_vlpazut363q5)

[**4. Thứ tự video 3**](#_fygteh75xm2k)

[**5. Các resource và ghi chú đính kèm: 5**](#_ov1crb3wvcbb)

[a. Thư viện cần cho Entity Framework Core 5](#_s8cacd7htqvh)

[b. Code lớp helper dùng cho mã hóa Hash (MD5, SHA512) 6](#_nc3fs5pf0gqt)

[c. Thư viện AutoMapper dùng để map đối tượng từ kiểu này sang kiểu khác 7](#_okpj6aivd7hc)

[d. Xác thực người dùng [Authentication] 7](#_jwc2jt7gscy3)

[e. Thanh toán Paypal (JS SDK) 7](#_iv7cknl87rhf)

[f. Thanh toán VNPAY 13](#_tpw9gdyaroev)

[\*\*\*Template tham khảo 13](#_sz31u2gv9vtg)

## 

## **1. Mục đích**

* Hệ thống các bước thực hiện dự án từ bắt đầu cho đến kết thúc.
* Củng cố lại kiến thức đã học trong quá trình của môn. Với website này, bạn sẽ được ôn tập và vận dụng phần lý thuyết cơ sở đã học để củng cố vững chắc kiến thức
* Xây dựng một thư viện kỹ thuật lập trình để tra cứu khi cần thiết.

## **2. Yêu cầu**

* Có kiến thức cơ bản về HTML, CSS, Bootstrap, jQuery (nếu có).
* Có kiến thức cơ bản về C#, ASP.NET Core, mô hình MVC.
* Có kiến thức cơ bản về cơ sở dữ liệu SQL Server.

## **3. Nội dung thực hiện**

1. **Phân tích và thiết kế website**
   1. Khởi tạo project (type: MVC)
   2. Dựng lại CSDL **HShop2023** (version SQL Server 2019)
   3. Chuẩn bị EF Core Database First
      1. Phát sinh Entity Model
      2. Khai báo chuỗi kết nối appsettings.json
      3. Đăng ký sử dụng **HShop2023Context**
      4. **Cài EF Core**
      5. **Cài EF Core.sqlServer**
      6. **Cài EF Core.Tools**
      7. **Version .net core nên = EF Core**
2. **Xây dựng Layout cho website**
   1. Layout trang chủ khách hàng - <https://youtu.be/g7WjoJOMHyI>
   2. Module hóa các thành phần giao diện
   3. Menu động sử dụng ViewComponent - <https://youtu.be/yTOcQtF1lzo>
   4. Đa ngôn ngữ
   5. Viết mã cho các module giao diện của layout trang chủ
      1. Quảng cáo
      2. Chủng loại
      3. Nhà cung cấp
      4. Hàng đặc biệt
      5. Hàng khuyến mại
3. Xây dựng trang hàng hóa
   1. Lọc, phân loại hàng hóa - <https://youtu.be/ocu5he0JSAs>
   2. Phân trang hàng hóa với ajax
   3. Tải bổ sung hàng hóa với ajax
4. Xây dựng trang chi tiết hàng hóa
   1. Thông tin chi tiết
   2. Hàng cùng loại
   3. Hàng cùng nhà cung cấp
   4. Hàng đã xem
5. Giỏ hàng
   1. Phân tích hoạt động của giỏ hàng (Shopping Cart)
   2. Điều khiển chọn hàng - <https://youtu.be/zooxDCy1oZY>
   3. Hiển thị thông tin tổng hợp giỏ hàng - <https://youtu.be/K01EacugLzo>
   4. Hiệu ứng bay vào giỏ / Hiệu ứng biến mất hàng bị xóa
   5. Quản lý giỏ hàng
6. Quản lý thành viên
   1. Đăng ký
   2. Đăng nhập
   3. Đăng xuất
   4. Quên mật khẩu
   5. Đổi mật khẩu
   6. Quản lý tài khoản
   7. Quản lý thông tin riêng tư
   8. Đăng nhập với tài khoản ngoài (Google, Facebook)
7. Đặt hàng
   1. Đặt hàng
   2. Thanh toán trực tuyến với Paypal/Strip, Momo/VNPay, …
   3. Quản lý đơn hàng
   4. Quản lý hàng hóa đã đặt
8. Send mail
   1. Đăng ký tài khoản - Active account
   2. Đổi mật khẩu
   3. Thông tin đặt hàng
   4. Quảng cáo
9. Một số tiện ích cho Web
   1. sweetalert2
   2. summernote
   3. captcha
   4. google map
   5. chartjs
10. Xây dựng layout quản trị
    1. Tạo Area Admin
    2. Xây dựng layout Admin
11. Quản lý hóa đơn
12. Quản lý loại
13. Quản lý hàng hóa
14. Thống kê số liệu kinh doanh
    1. Hàng tồn kho
    2. Doanh số của từng mặt hàng
    3. Doanh số của từng loại hàng
    4. Doanh số của từng nhà cung cấp
    5. Doanh số bán cho từng khách hàng
    6. Doanh số của từng tháng
    7. Doanh số của từng quý
    8. Doanh số của từng năm
15. Phân quyền sử dụng
    1. Quản lý tài khoản
    2. Quản lý vai trò
    3. Phân quyền

## **4. Thứ tự video**

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Link | Tiêu đề video | Nội dung |
| 1 | <https://youtu.be/7hwegNKkq9k> | 00 - Giới thiệu | Giới thiệu dự án, các chức năng sẽ thực hiện |
| 2 | <https://youtu.be/JNFBVt2tbCs> | 01 - Khởi tạo Project | - Khởi tạo Project  - Tạo project dạng Model - View - Controller  - Dựng lại Database trong SQL Server  - Phát sinh Entity Model (Database First)  - Đăng ký DbContext  - Thử tạo Controller with View, using Entity Framework Core |
| 3 | <https://youtu.be/g7WjoJOMHyI> | 02 - Layout trang chủ | - Tạo mới Layout Template (Razor Layout)  - Thêm các thành phần trên template: Header, Footer, Content |
| 4 | <https://youtu.be/yTOcQtF1lzo> | 03 - Tạo Menu hàng hóa | Sử dụng ViewComponent để tạo menu danh mục trang hàng hóa. |
| 5 | <https://youtu.be/ocu5he0JSAs> | 04 - Trang Danh sách hàng hóa | - Lấy danh sách hàng hóa (tất cả, theo loại)  - Lấy danh sách hàng hóa từ tìm kiếm theo tên  - Tạo layout template từ layout template đã có (nested layout) |
| 6 | <https://youtu.be/rTLZTVyGwlw> | 05 - Chi tiết hàng hóa |  |
| 7 | <https://youtu.be/zooxDCy1oZY> | 06 - Đưa hàng vào giỏ |  |
| 8 | <https://youtu.be/K01EacugLzo> | 07 - Thông tin Giỏ hàng |  |
| 9 | <https://youtu.be/uuMMw2If1js> | 08 - Chức năng Đăng ký Thành viên | - Tạo ViewModel đăng ký thành viên  - Tạo trang đăng ký thành viên  - AutoMapper  - Xử lý đăng ký (mật khẩu mã hóa hash kết hợp salt key) |
| 10 | <https://youtu.be/JUkop0oeKMg> | 09 - Chức năng Đăng nhập | Định nghĩa Login ViewModel  Khai báo xác thực (Authentication) người dùng  Thiết kế view đăng nhập  Xử lý đăng nhập  Xử lý đăng xuất |
| 11 | <https://youtu.be/ZORr_H5QOY4> | 10 - Thanh toán COD |  |
| 12 |  | 11 - Thanh toán Paypal |  |

## **5. Các resource và ghi chú đính kèm:**

### ***a. Thư viện cần cho Entity Framework Core***

Nhớ cài 3 package sau khi làm việc với Entity Framework Core:

* Microsoft.EntityFrameworkCore
* Microsoft.EntityFrameworkCore.SqlServer
* Microsoft.EntityFrameworkCore.Tools

*Ngoài ra, các bạn chú ý version .NET Core = version Entity Framework Core (vd: cùng là v8)*

### ***b. Code lớp helper dùng cho mã hóa Hash (MD5, SHA512)***

using System.Security.Cryptography;

using System.Text;

namespace MyCommerce.Models

{

public static class DataEncryptionExtensions

{

#region [Hashing Extension]

public static string ToSHA256Hash(this string password, string? saltKey)

{

var sha256 = SHA256.Create();

byte[] encryptedSHA256 = sha256.ComputeHash(Encoding.UTF8.GetBytes(string.Concat(password, saltKey)));

sha256.Clear();

return Convert.ToBase64String(encryptedSHA256);

}

public static string ToSHA512Hash(this string password, string? saltKey)

{

SHA512Managed sha512 = new SHA512Managed();

byte[] encryptedSHA512 = sha512.ComputeHash(Encoding.UTF8.GetBytes(string.Concat(password, saltKey)));

sha512.Clear();

return Convert.ToBase64String(encryptedSHA512);

}

public static string ToMd5Hash(this string password, string? saltKey)

{

using (var md5 = MD5.Create())

{

byte[] data = md5.ComputeHash(Encoding.UTF8.GetBytes(string.Concat(password, saltKey)));

StringBuilder sBuilder = new StringBuilder();

for (int i = 0; i < data.Length; i++)

{

sBuilder.Append(data[i].ToString("x2"));

}

return sBuilder.ToString();

}

}

#endregion

}

}

Cách sử dụng: **<chuoi\_password>.ToMD5Hash()** hoặc **<chuoi\_password>.ToMD5Hash(“salt\_key”)**

### ***c. Thư viện AutoMapper dùng để map đối tượng từ kiểu này sang kiểu khác***

* Cần cài package: AutoMapper.Extensions.Microsoft.DependencyInjection
* Định nghĩa lớp Map giữa 2 kiểu dữ liệu:  
  public class AutoMapperProfile : Profile

{

public AutoMapperProfile()

{

CreateMap<RegisterVM, KhachHang>(); }

}

* Đăng ký sử dụng ở Program.cs:  
  *builder.Services.AddAutoMapper(typeof(AutoMapperProfile));*

Tham khảo thêm: <https://docs.automapper.org/en/stable/Dependency-injection.html>

### **d. Xác thực người dùng [Authentication]**

Có 2 cách xác thực: Cookie hay Token. Trong video này mình dùng Cookie cho web, bạn tham khảo thêm trên trang Microsoft: <https://learn.microsoft.com/en-us/aspnet/core/security/authentication/cookie?view=aspnetcore-8.0>

Sau khi xác thực thành công, bạn muốn protect action hay controller nào thì thêm thuộc tính [Authorize] vào trước action/controller đó.

### ***e. Thanh toán Paypal (JS SDK)***

**Chuẩn bị:**

* Đăng ký tài khoản Paypal
* Vào <https://developer.paypal.com/>
* Tạo tài khoản sandbox để demo: [Sandbox accounts - PayPal Developer](https://developer.paypal.com/dashboard/accounts) (gồm 2 loại Personal - người mua, Business - người bán)
* Tạo mới App ở mục **Apps & Credentials** [Applications - PayPal Developer](https://developer.paypal.com/dashboard/applications/sandbox) hoặc đứng ngay trong tài khoản Business để tạo.  
  Sau đó copy App ID, App secret key để dành để thêm vào project.

Tham khảo: <https://developer.paypal.com/sdk/js>

Ở trên trang Paypal docs toàn hướng dẫn cho JS không nên mình chuẩn bị sẵn cho các bạn một số lệnh:

Định nghĩa lớp **PaypalClient** để xác thực và tạo đơn hàng:

public sealed class PaypalClient

{

public string Mode { get; }

public string ClientId { get; }

public string ClientSecret { get; }

public string BaseUrl => Mode == "Live"

? "https://api-m.paypal.com"

: "https://api-m.sandbox.paypal.com";

public PaypalClient(string clientId, string clientSecret, string mode)

{

ClientId = clientId;

ClientSecret = clientSecret;

Mode = mode;

}

private async Task<AuthResponse> Authenticate()

{

var auth = Convert.ToBase64String(Encoding.UTF8.GetBytes($"{ClientId}:{ClientSecret}"));

var content = new List<KeyValuePair<string, string>>

{

new("grant\_type", "client\_credentials")

};

var request = new HttpRequestMessage

{

RequestUri = new Uri($"{BaseUrl}/v1/oauth2/token"),

Method = HttpMethod.Post,

Headers =

{

{ "Authorization", $"Basic {auth}" }

},

Content = new FormUrlEncodedContent(content)

};

var httpClient = new HttpClient();

var httpResponse = await httpClient.SendAsync(request);

var jsonResponse = await httpResponse.Content.ReadAsStringAsync();

var response = JsonSerializer.Deserialize<AuthResponse>(jsonResponse);

return response;

}

public async Task<CreateOrderResponse> CreateOrder(string value, string currency, string reference)

{

var auth = await Authenticate();

var request = new CreateOrderRequest

{

intent = "CAPTURE",

purchase\_units = new List<PurchaseUnit>

{

new()

{

reference\_id = reference,

amount = new Amount

{

currency\_code = currency,

value = value

}

}

}

};

var httpClient = new HttpClient();

httpClient.DefaultRequestHeaders.Authorization = AuthenticationHeaderValue.Parse($"Bearer {auth.access\_token}");

var httpResponse = await httpClient.PostAsJsonAsync($"{BaseUrl}/v2/checkout/orders", request);

var jsonResponse = await httpResponse.Content.ReadAsStringAsync();

var response = JsonSerializer.Deserialize<CreateOrderResponse>(jsonResponse);

return response;

}

public async Task<CaptureOrderResponse> CaptureOrder(string orderId)

{

var auth = await Authenticate();

var httpClient = new HttpClient();

httpClient.DefaultRequestHeaders.Authorization = AuthenticationHeaderValue.Parse($"Bearer {auth.access\_token}");

var httpContent = new StringContent("", Encoding.Default, "application/json");

var httpResponse = await httpClient.PostAsync($"{BaseUrl}/v2/checkout/orders/{orderId}/capture", httpContent);

var jsonResponse = await httpResponse.Content.ReadAsStringAsync();

var response = JsonSerializer.Deserialize<CaptureOrderResponse>(jsonResponse);

return response;

}

}

public sealed class AuthResponse

{

public string scope { get; set; }

public string access\_token { get; set; }

public string token\_type { get; set; }

public string app\_id { get; set; }

public int expires\_in { get; set; }

public string nonce { get; set; }

}

public sealed class CreateOrderRequest

{

public string intent { get; set; }

public List<PurchaseUnit> purchase\_units { get; set; } = new();

}

public sealed class CreateOrderResponse

{

public string id { get; set; }

public string status { get; set; }

public List<Link> links { get; set; }

}

public sealed class CaptureOrderResponse

{

public string id { get; set; }

public string status { get; set; }

public PaymentSource payment\_source { get; set; }

public List<PurchaseUnit> purchase\_units { get; set; }

public Payer payer { get; set; }

public List<Link> links { get; set; }

}

public sealed class PurchaseUnit

{

public Amount amount { get; set; }

public string reference\_id { get; set; }

public Shipping shipping { get; set; }

public Payments payments { get; set; }

}

public sealed class Payments

{

public List<Capture> captures { get; set; }

}

public sealed class Shipping

{

public Address address { get; set; }

}

public class Capture

{

public string id { get; set; }

public string status { get; set; }

public Amount amount { get; set; }

public SellerProtection seller\_protection { get; set; }

public bool final\_capture { get; set; }

public string disbursement\_mode { get; set; }

public SellerReceivableBreakdown seller\_receivable\_breakdown { get; set; }

public DateTime create\_time { get; set; }

public DateTime update\_time { get; set; }

public List<Link> links { get; set; }

}

public class Amount

{

public string currency\_code { get; set; }

public string value { get; set; }

}

public sealed class Link

{

public string href { get; set; }

public string rel { get; set; }

public string method { get; set; }

}

public sealed class Name

{

public string given\_name { get; set; }

public string surname { get; set; }

}

public sealed class SellerProtection

{

public string status { get; set; }

public List<string> dispute\_categories { get; set; }

}

public sealed class SellerReceivableBreakdown

{

public Amount gross\_amount { get; set; }

public PaypalFee paypal\_fee { get; set; }

public Amount net\_amount { get; set; }

}

public sealed class Paypal

{

public Name name { get; set; }

public string email\_address { get; set; }

public string account\_id { get; set; }

}

public sealed class PaypalFee

{

public string currency\_code { get; set; }

public string value { get; set; }

}

public class Address

{

public string address\_line\_1 { get; set; }

public string address\_line\_2 { get; set; }

public string admin\_area\_2 { get; set; }

public string admin\_area\_1 { get; set; }

public string postal\_code { get; set; }

public string country\_code { get; set; }

}

public sealed class Payer

{

public Name name { get; set; }

public string email\_address { get; set; }

public string payer\_id { get; set; }

}

public sealed class PaymentSource

{

public Paypal paypal { get; set; }

}

Phần appsettings.json thêm phần config dành cho Paypal:

"PaypalOptions": {

"Mode": "Sandbox",

"ClientId": "id",

"ClientSecret": "secret"

},

Khai báo sử dụng lớp PaypalClient ở Program.cs:

đăng ký PaypalClient dạng Singleton (có 1 thể hiện/instance duy nhất trong toàn ứng dụng)

builder.Services.AddSingleton(x =>

new PaypalClient(

builder.Configuration["PayPalOptions:ClientId"],

builder.Configuration["PayPalOptions:ClientSecret"],

builder.Configuration["PayPalOptions:Mode"]

)

);

Thực hiện thanh toán:

* Frontend: Hiện nút PAYPAL và khai báo các sự kiện createOrder, captureOrder
* Backend: Viết API cho thao tác CreateOrder và CaptureOrder.

### ***f. Thanh toán VNPAY***

[Hướng dẫn tích hợp Cổng thanh toán VNPAY · Cổng thanh toán VNPAY (vnpayment.vn)](https://sandbox.vnpayment.vn/apis/docs/huong-dan-tich-hop/)

[Kết nối Cổng thanh toán VNPAY · Cổng thanh toán VNPAY (vnpayment.vn)](https://sandbox.vnpayment.vn/apis/)

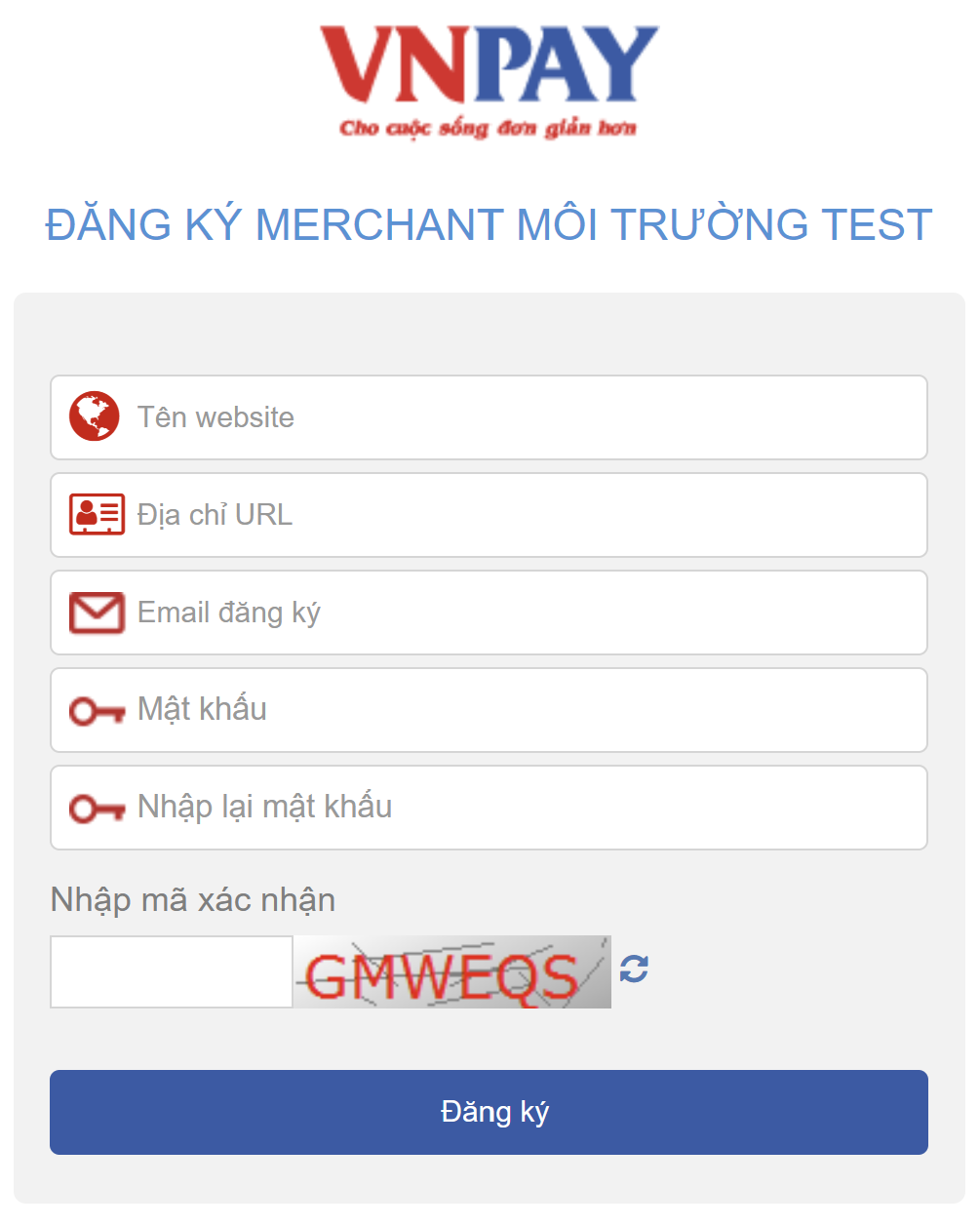
VNPAY cung cấp sẵn ví dụ nhưng C# WebForm: [Download · Cổng thanh toán VNPAY (vnpayment.vn)](https://sandbox.vnpayment.vn/apis/downloads/)

Thông tin thẻ demo: [Demo Cổng thanh toán VNPAY · Cổng thanh toán VNPAY (vnpayment.vn)](https://sandbox.vnpayment.vn/apis/vnpay-demo/)

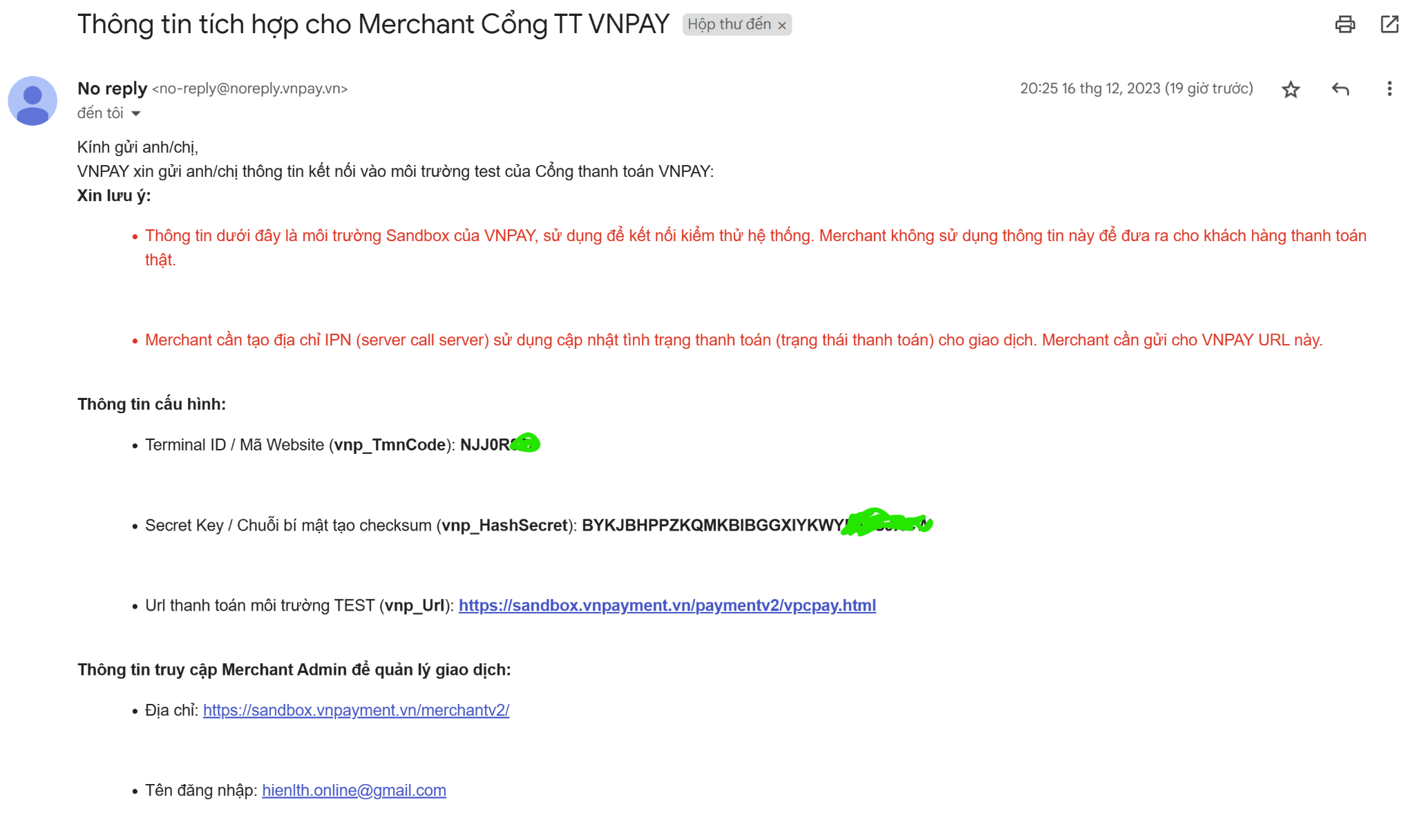
Bảng mã lỗi khi thanh toán: [Bảng mã lỗi · Cổng thanh toán VNPAY (vnpayment.vn)](https://sandbox.vnpayment.vn/apis/docs/bang-ma-loi/)

***Đăng ký***

Đăng ký tài khoản VNPAY tại địa chỉ: <https://sandbox.vnpayment.vn/devreg/>



Sau đó bạn check email để kích hoạt tài khoản và nhận 01 email chứa thông tin tích hợp:



***Tích hợp***

Thông tin lớp **VnPayLibrary** (lấy từ ví dụ mẫu nhưng có chỉnh sửa):

public class VnPayLibrary

{

private readonly SortedList<string, string> \_requestData = new SortedList<string, string>(new VnPayCompare());

private readonly SortedList<string, string> \_responseData = new SortedList<string, string>(new VnPayCompare());

public void AddRequestData(string key, string value)

{

if (!string.IsNullOrEmpty(value))

{

\_requestData.Add(key, value);

}

}

public void AddResponseData(string key, string value)

{

if (!string.IsNullOrEmpty(value))

{

\_responseData.Add(key, value);

}

}

public string GetResponseData(string key)

{

return \_responseData.TryGetValue(key, out var retValue) ? retValue : string.Empty;

}

#region Request

public string CreateRequestUrl(string baseUrl, string vnpHashSecret)

{

var data = new StringBuilder();

foreach (var (key, value) in \_requestData.Where(kv => !string.IsNullOrEmpty(kv.Value)))

{

data.Append(WebUtility.UrlEncode(key) + "=" + WebUtility.UrlEncode(value) + "&");

}

var querystring = data.ToString();

baseUrl += "?" + querystring;

var signData = querystring;

if (signData.Length > 0)

{

signData = signData.Remove(data.Length - 1, 1);

}

var vnpSecureHash = Utils.HmacSHA512(vnpHashSecret, signData);

baseUrl += "vnp\_SecureHash=" + vnpSecureHash;

return baseUrl;

}

#endregion

#region Response process

public bool ValidateSignature(string inputHash, string secretKey)

{

var rspRaw = GetResponseData();

var myChecksum = Utils.HmacSHA512(secretKey, rspRaw);

return myChecksum.Equals(inputHash, StringComparison.InvariantCultureIgnoreCase);

}

private string GetResponseData()

{

var data = new StringBuilder();

if (\_responseData.ContainsKey("vnp\_SecureHashType"))

{

\_responseData.Remove("vnp\_SecureHashType");

}

if (\_responseData.ContainsKey("vnp\_SecureHash"))

{

\_responseData.Remove("vnp\_SecureHash");

}

foreach (var (key, value) in \_responseData.Where(kv => !string.IsNullOrEmpty(kv.Value)))

{

data.Append(WebUtility.UrlEncode(key) + "=" + WebUtility.UrlEncode(value) + "&");

}

//remove last '&'

if (data.Length > 0)

{

data.Remove(data.Length - 1, 1);

}

return data.ToString();

}

#endregion

}

public class Utils

{

public static string HmacSHA512(string key, string inputData)

{

var hash = new StringBuilder();

var keyBytes = Encoding.UTF8.GetBytes(key);

var inputBytes = Encoding.UTF8.GetBytes(inputData);

using (var hmac = new HMACSHA512(keyBytes))

{

var hashValue = hmac.ComputeHash(inputBytes);

foreach (var theByte in hashValue)

{

hash.Append(theByte.ToString("x2"));

}

}

return hash.ToString();

}

// có chế biến cho .NET Core MVC

public static string GetIpAddress(HttpContext context)

{

var ipAddress = string.Empty;

try

{

var remoteIpAddress = context.Connection.RemoteIpAddress;

if (remoteIpAddress != null)

{

if (remoteIpAddress.AddressFamily == AddressFamily.InterNetworkV6)

{

remoteIpAddress = Dns.GetHostEntry(remoteIpAddress).AddressList

.FirstOrDefault(x => x.AddressFamily == AddressFamily.InterNetwork);

}

if (remoteIpAddress != null) ipAddress = remoteIpAddress.ToString();

return ipAddress;

}

}

catch (Exception ex)

{

return "Invalid IP:" + ex.Message;

}

return "127.0.0.1";

}

}

public class VnPayCompare : IComparer<string>

{

public int Compare(string x, string y)

{

if (x == y) return 0;

if (x == null) return -1;

if (y == null) return 1;

var vnpCompare = CompareInfo.GetCompareInfo("en-US");

return vnpCompare.Compare(x, y, CompareOptions.Ordinal);

}

}

Mô tả lớp kết quả trả về khi thanh toán:

public class **VnPaymentResponseModel**

{

public bool Success { get; set; }

public string PaymentMethod { get; set; }

public string OrderDescription { get; set; }

public string OrderId { get; set; }

public string PaymentId { get; set; }

public string TransactionId { get; set; }

public string Token { get; set; }

public string VnPayResponseCode { get; set; }

}

### **\*\*\*Template tham khảo**

* Template trang checkout: <https://bootdey.com/snippets/tagged/checkout>
* Template customer đang sử dụng: <https://htmlcodex.com/vegetable-website-template/>
* Template có thể tìm ở đây (có tiền mua thì dễ, không thì lựa chọn BootStrap 5 free nhé):
  + <https://htmlcodex.com/template/ecommerce/>
  + <https://www.creative-tim.com/bootstrap-themes/free>

—------------------------------------------------------------------------------------------------------------------------

*Nếu các bạn thấy hay, vui lòng Like, Share, comment và giới thiệu đến nhiều người hơn.*

*Ủng hộ tôi 1 ly cafe để có tinh thần sảng khoái làm nhiều video hơn:*

***BIDV 0559.667.467***

***XIN CẢM ƠN!***