

Nhóm 5:

Phan Quốc Khánh

Dương Công Phước

Nguyễn Thành Thi

Lê Quốc Việt

Lưu Hoàng Phương Nam

Câu 4: Đối ABC thuê công ty XYZ thiết kế một Website bán mô hình figure(mô hình đồ chơi). Hãy giúp công ty XYZ thiết kế phần mềm này bằng việc xây dựng các thành phần sau:

a. Hệ thống các màn hình giao diện

giao diện đăng nhập đăng ký

LOGIN TO YOUR ACCOUNT

Username

nam

Password

LOGIN

Login by Google

Register

Forgot Password

ĐĂNG KÝ NGƯỜI DÙNG MỚI!

Tên người dùng

Enter your username

Email

Enter your email

Mật khẩu

Enter your password

TẠO

FORGET PASSWORD

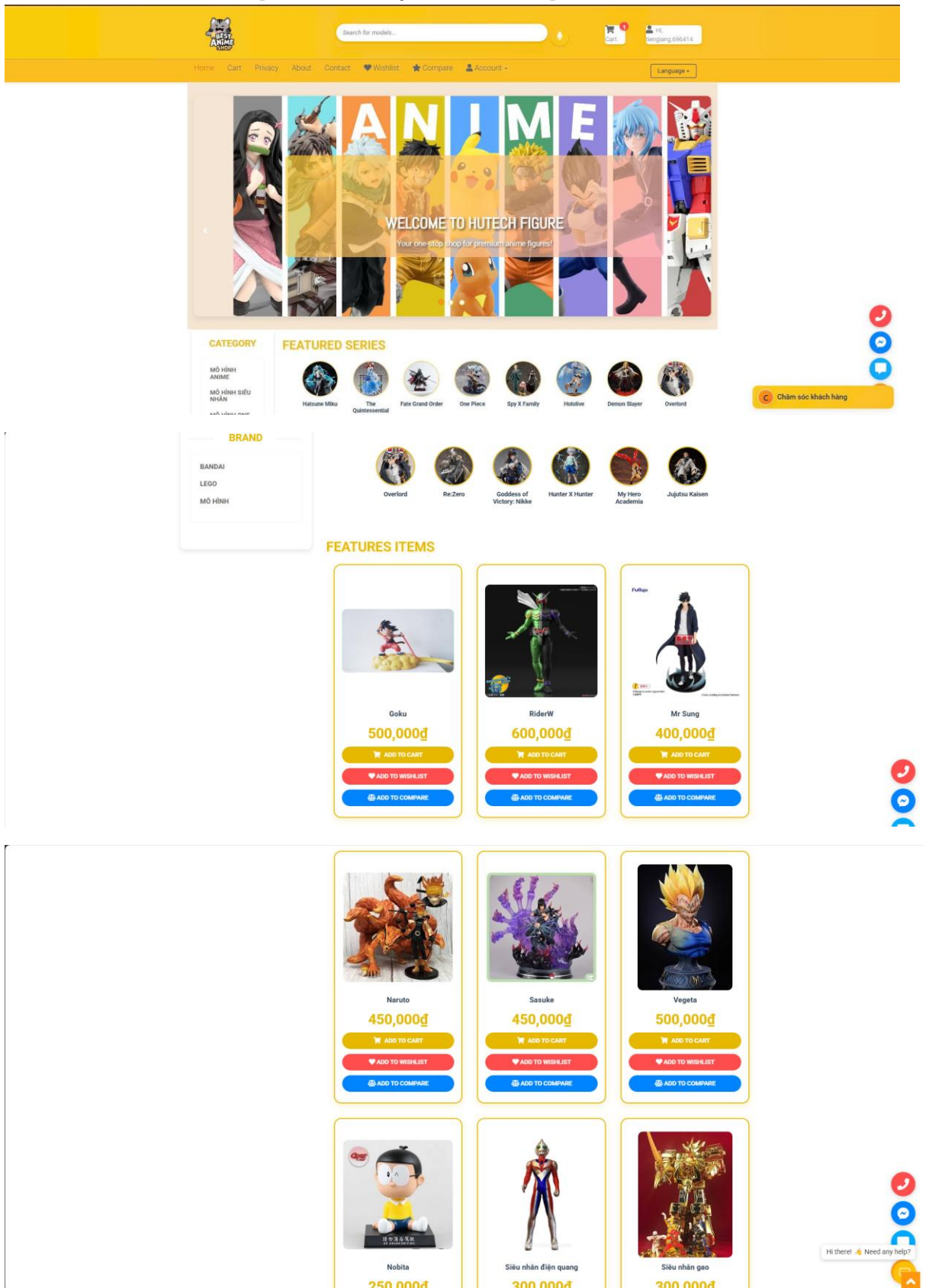
Email

Enter your email

SEND

Đăng ký

giao diện trang chủ





Shin

350,000đ

ADD TO CART

ADD TO WISHLIST

ADD TO COMPARE

VIDEO REVIEWS



MÔ HÌNH GOJO X SUKUNA LUMINASTA (SEGA)

Review mô hình Gojo và Sukuna từ Jujutsu Kaisen, phiên bản Luminasta của SEGA...



MÔ HÌNH 2B - ICHIBAN KUJI NIER: AUTOMATA VER1.1A

Review mô hình 2B từ NieR:Automata, phiên bản Ichiban Kuji Ver1.1a...



MÔ HÌNH HATSUNE MIKU - COREFUL - LOLITA VER.


Review mô hình Hatsune Miku phiên bản Coreful Lolita của TAITO...

FAMOUS BRANDS



NEWS


10/03/2025



[ELCOCO/SYSTEM SERVICE/KONAMI]

Hutech Figure update list mô hình Game Prize sắp ra mắt trong tháng 3/2025 của hãng ELCOCO/SYSTEM SERVICE/KONAMI. Giá dao động từ 350K - 750K. Các bạn có thể...


10/03/2025



[FURYU] LỊCH PHÁT HÀNH MỚI

Hutech Figure update list mô hình Game Prize sắp ra mắt trong tháng 3/2025 của hãng FURYU. Giá dao động từ 350K - 750K. Các bạn có thể...


10/03/2025



[TAITO] LỊCH PHÁT HÀNH MỚI

Hutech Figure update list mô hình Game Prize sắp ra mắt trong tháng 3/2025 của hãng TAITO. Giá dao động từ 350K - 750K. Các bạn có thể...

10/03/2025



[ICHIBAN KUJI] LỊCH PHÁT HÀNH MỚI

Hutech Figure update list mô hình Ichiban Kuji sắp ra mắt trong tháng 3/2025 của hãng BANDAI. Các bạn có thể đặt trước...

REVIEWS

Mô hình Gintama mình mua đẹp không tí vết, đóng gói rất cẩn thận! Rất hài lòng với sản phẩm này, sẽ quay lại mua thêm!

★★★★★

Nguyễn Minh Anh

Mình đã mua một chiếc vòng cổ anime ở đây, chất lượng tuyệt vời và giao hàng rất nhanh. Cảm ơn shop nhiều nhé!

★★★★★

Trần Quốc Bảo

Mô hình đến đúng hạn, hộp không bị móp méo gì cả. Chất lượng sản phẩm thì khỏi chê, rất đáng tiền. Sẽ ủng hộ shop lâu dài!

★★★★★

Lê Thị Hồng Nhung

giao diện giỏ hàng và thanh toán


HÌNH ẢNH

TÊN SẢN PHẨM

GIÁ TIỀN

SỐ LƯỢNG

THÀNH TIỀN



Bocchi

650.000 VNĐ

+

2

-

1.300.000 VNĐ

X

Grand Total: 1.300.000 VNĐ

Coupon Code:

Apply

Shipping Cost: 0 VNĐ Xóa phí ship

Họ tên

Số điện thoại

Tỉnh thành phố

Quận huyện

Phường xã

Số nhà, đường

Tính phí ship

Xóa tất cả

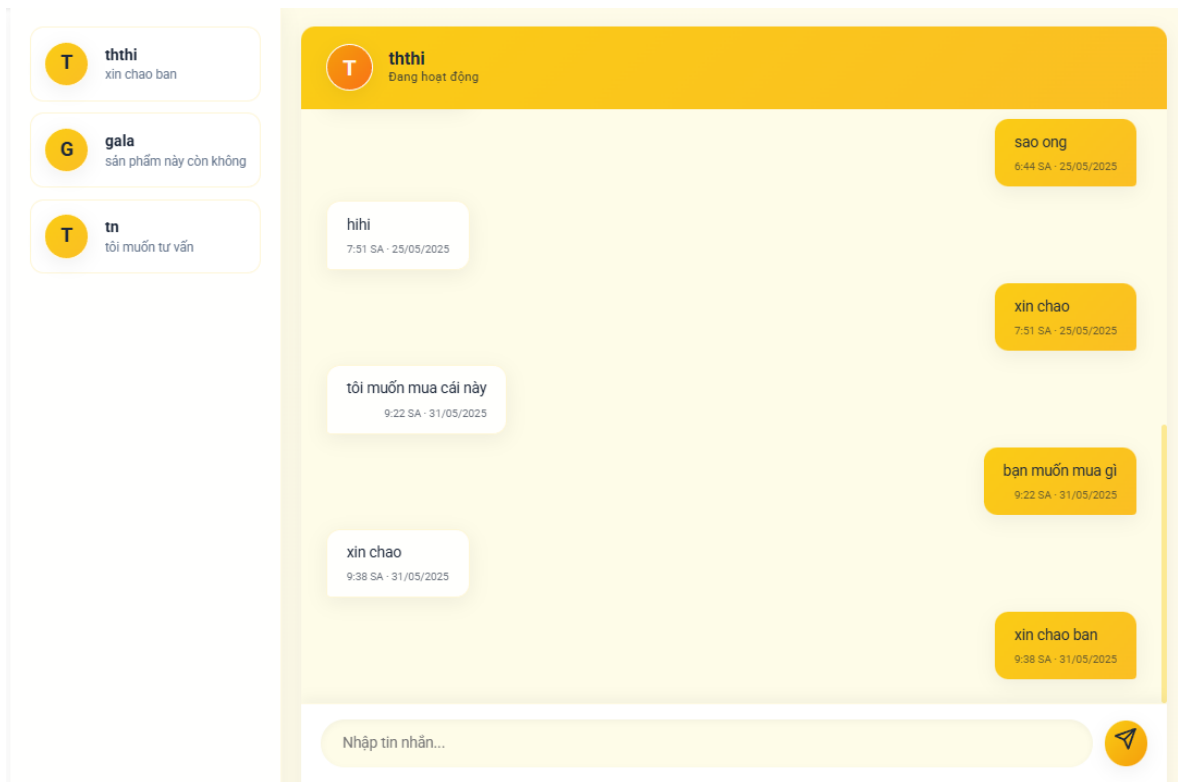
Vui lòng nhập thông tin nơi ở để tính phí vận chuyển.

Thanh toán

Pay with MoMo

Pay with VNPAY

giao diện chăm sóc khách hàng



b. Hệ thống các hàm xử lý

Hàm đăng ký :

```
public async Task<IActionResult> Create(UserModel user)
```

```

{
    if (ModelState.IsValid)
    {
        AppUserModel newUser = new AppUserModel
        {
            UserName = user.UserName,
            Email = user.Email
        };

        IdentityResult result = await _userManager.CreateAsync(newUser, user.Password);
        if (result.Succeeded)
        {
            // Gán role mặc định là "CUSTOMER"
            await _userManager.AddToRoleAsync(newUser, "CUSTOMER");

            TempData["success"] = "Đăng ký tài khoản thành công";
            return Redirect("/account/login");
        }

        foreach (IdentityError error in result.Errors)
        {
            ModelState.AddModelError("", error.Description);
        }
    }

    return View(user);
}

```

Hàm đăng nhập :

```

[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Login(LoginViewModel loginVM)
{
    if (ModelState.IsValid)

```

```

{
    Microsoft.AspNetCore.Identity.SignInResult result = await
_signInManager.PasswordSignInAsync(loginVM.UserName, loginVM.Password, false,
false);

    if (result.Succeeded)
    {
        // Lấy thông tin người dùng
        var user = await _userManager.FindByNameAsync(loginVM.UserName);

        if (user != null)
        {
            // Lấy danh sách vai trò
            var roles = await _userManager.GetRolesAsync(user);

            // Chuyển hướng dựa trên vai trò
            if (roles.Contains("CSKH"))
            {
                return RedirectToAction("Messages", "Messages");
            }
            else if (roles.Contains("ADMIN") || roles.Contains("EMPLOYEE"))
            {
                return RedirectToAction("Index", "Admin");
            }
            else if (roles.Contains("CUSTOMER"))
            {
                return RedirectToAction("Index", "Home");
            }
            else
            {
                // Vai trò mặc định
                return Redirect(loginVM.ReturnUrl ?? "/");
            }
        }
    }

    // Nếu không tìm thấy người dùng, chuyển hướng mặc định

```

```

        return Redirect(loginVM.ReturnUrl ?? "/");
    }

    ModelState.AddModelError("", "Invalid Username and Password");
}

return View(loginVM);
}

```

Hàm thêm sản phẩm vào giỏ hàng :

[HttpPost]

public async Task<ActionResult> Add(long Id)

```

{
    ProductModel product = await _dataContext.Products.FindAsync(Id);
    if (product == null)
    {
        return Json(new { success = false, message = "Sản phẩm không tồn tại" });
    }
}

```

```

List<CartItemModel> cart = HttpContext.Session.GetJson<List<CartItemModel>>("Cart")
?? new List<CartItemModel>();

```

```

CartItemModel cartItem = cart.Where(c => c.ProductId == Id).FirstOrDefault();

```

```

if (cartItem == null)

```

```

{
    cart.Add(new CartItemModel(product));
}

```

```

else

```

```

{
    cartItem.Quantity += 1;
}

```

```

HttpContext.Session.SetJson("Cart", cart);

```

```

int totalCount = cart.Sum(x => x.Quantity);

```



```

        return Json(new { success = true, count = totalCount, message = "Thêm vào giỏ hàng thành công" });
    }

```

Hàm tính phí ship :

```

public async Task<ActionResult> GetShipping(ShippingModel shippingModel, string tinh,
string quan, string phuong, string houseNumber, string fullName, string phoneNumber)
{

```

```

    var existingShipping = await _dbContext.Shippings
        .FirstOrDefaultAsync(x => x.City == tinh && x.District == quan && x.Ward == phuong);

```

```

    decimal shippingPrice = existingShipping?.Price ?? 50000;

```

```

    var shippingPriceJson = JsonConvert.SerializeObject(shippingPrice);

```

```

    var cookieOptions = new CookieOptions

```

```

    {
        HttpOnly = true,
        Expires = DateTimeOffset.UtcNow.AddMinutes(30),
        Secure = true

```

```

    };
    Response.Cookies.Append("ShippingPrice", shippingPriceJson, cookieOptions);

```

```

    var address = new AddressModel

```

```

    {
        City = tinh,
        District = quan,
        Ward = phuong,
        HouseNumber = houseNumber,
        FullName = fullName,
        PhoneNumber = phoneNumber

```

```

    };
    HttpContext.Session.SetJson("ShippingAddress", address);

```

```

    return Json(new

```

```

{
    success = true,
    shippingPrice = shippingPrice.ToString("#,##0 VNĐ"),
    address = address
});
}

```

Hàm thanh toán :

```
public async Task<IActionResult> Checkout(string PaymentMethod, string PaymentId)
```

```

{
    var userEmail = User.FindFirstValue(ClaimTypes.Email);
    if (userEmail == null)
    {
        return RedirectToAction("Login", "Account");
    }

    var orderCode = Guid.NewGuid().ToString();
    var shippingPriceCookie = Request.Cookies["ShippingPrice"];
    double shippingPrice = shippingPriceCookie != null ?
    JsonConvert.DeserializeObject<double>(shippingPriceCookie) : 0;
    var coupon_code = Request.Cookies["CouponTitle"];

```

```
// Lấy thông tin địa chỉ từ Session
```

```

var address = HttpContext.Session.GetJson<AddressModel>("ShippingAddress");
if (address == null)
{
    TempData["error"] = "Vui lòng nhập thông tin địa chỉ giao hàng.";
    return RedirectToAction("Index", "Cart");
}

```

```
var orderItem = new OrderModel
```

```

{
    OrderCode = orderCode,
    ShippingCost = shippingPrice,

```

```

        CouponCode = coupon_code,
        UserName = userEmail,
        Status = 1,
        CreatedDate = DateTime.Now,
        PaymentMethod = PaymentMethod + " " + PaymentId,
        City = address.City,
        District = address.District,
        Ward = address.Ward,
        HouseNumber = address.HouseNumber,
        FullName = address.FullName, // Thêm FullName
        PhoneNumber = address.PhoneNumber
    };

    _dataContext.Add(orderItem);
    _dataContext.SaveChanges();

    List<CartItemModel> cartItems =
HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();
    double total = 0;
    List<OrderDetails> orderDetailsList = new List<OrderDetails>();

    foreach (var cart in cartItems)
    {
        var orderDetails = new OrderDetails
        {
            UserName = userEmail,
            OrderCode = orderCode,
            ProductId = cart.ProductId,
            Price = cart.Price,
            Quantity = cart.Quantity
        };

        var product = await _dataContext.Products.Where(p => p.Id ==
cart.ProductId).FirstAsync();

```

```

        product.Quantity -= cart.Quantity;
        product.Sold += cart.Quantity;
        _dataContext.Update(product);

        orderDetailsList.Add(orderDetails);
        _dataContext.Add(orderDetails);
        _dataContext.SaveChanges();

        total += cart.Price * cart.Quantity;
    }

    HttpContext.Session.Remove("Cart");
    Response.Cookies.Delete("ShippingPrice");
    HttpContext.Session.Remove("ShippingAddress");

    var emailBody = $"
<!DOCTYPE html>
<html>
<body style='font-family:Arial, sans-serif;'>
    <div style='max-width:600px; margin:0 auto;'>
        <h2 style='color:#2d89ef;'>Đặt hàng thành công!</h2>
        <p>Họ tên: {address.FullName}</p>
        <p>Số điện thoại: {address.PhoneNumber}</p>
        <p>Mã đơn hàng: <span style='color:#d9534f;'>{orderCode}</span></p>
        <p>Địa chỉ giao hàng: {address.HouseNumber}, {address.Ward}, {address.District},
        {address.City}</p>
        <table style='width:100%;'>
            <tr style='background-color:#2d89ef; color:#ffffff;'>
                <th>Sản phẩm</th><th>Giá</th><th>Số lượng</th><th>Thành tiền</th>
            </tr>";

    foreach (var item in orderDetailsList)
    {
        var product = _dataContext.Products.FirstOrDefault(p => p.Id == item.ProductId);

```

```

        if (product != null)
        {
            emailBody += $@"
            <tr>
                <td>{product.Name}</td>
                <td>{item.Price:C}</td>
                <td>{item.Quantity}</td>
                <td>{(item.Price * item.Quantity):C}</td>
            </tr>";
        }
    }
}

```

```

emailBody += $@"
    </table>
    <h3>Tổng tiền: {total + shippingPrice:C}</h3>
</div>
</body>
</html>";

```

```

        await _emailSender.SendEmailAsync(userEmail, "Xác nhận đơn hàng thành công",
        emailBody, true);

        TempData["success"] = "Checkout thành công, vui lòng chờ duyệt đơn hàng!";
        return RedirectToAction("History", "Account");
    }
}

```

Hàm thanh toán bằng momo và VNPay :

```

[HttpGet]
public async Task<ActionResult> PaymentCallBack(MomoInfoModel model)
{
    var response = _momoService.PaymentExecuteAsync(HttpContext.Request.Query);
    var requestQuery = HttpContext.Request.Query;
    if (requestQuery["errorCode"] == "0")
    {
        var newMomoInsert = new MomoInfoModel
    }
}

```

```

{
    OrderId = requestQuery["orderId"],
    FullName = User.FindFirstValue(ClaimTypes.Email),
    Amount = decimal.Parse(requestQuery["Amount"]),
    OrderInfo = requestQuery["orderInfo"],
    DatePaid = DateTime.Now
};

_dataContext.Add(new MomolInsert);
await _dataContext.SaveChangesAsync();
var PaymentMethod1 = "MOMO";
await Checkout(PaymentMethod1, requestQuery["orderId"]);
}
else
{
    TempData["error"] = "Đã hủy giao dịch Momo.";
    return RedirectToAction("Index", "Cart");
}
return View(response);
}

```

[HttpGet]

```

public async Task<ActionResult> PaymentCallbackVnpay()
{
    var response = _vnPayService.PaymentExecute(Request.Query);

    if (response.VnPayResponseCode == "00")
    {
        var newVnpayInsert = new VnpayModel
        {
            OrderId = response.OrderId,
            PaymentMethod = response.PaymentMethod,

```

```

        OrderDescription = response.OrderDescription,
        TransactionId = response.TransactionId,
        PaymentId = response.PaymentId,
        DateCreated = DateTime.Now
    };
    _dataContext.Add(newVnpayInsert);
    await _dataContext.SaveChangesAsync();
    var PaymentMethod = response.PaymentMethod;
    var PaymentId = response.PaymentId;
    await Checkout(PaymentMethod, PaymentId);
}
else
{
    TempData["error"] = "Đã hủy giao dịch Vnpay.";
    return RedirectToAction("Index", "Cart");
}
return View(response);
}
}

```

Hàm chat :

```

public async Task<ActionResult> Messages(string selectedUserId)
{
    var users = await _messageService.GetUsers();
    var currentUserId = _currentUserService.UserId;

    if (string.IsNullOrEmpty(selectedUserId))
    {
        var roles = await _context.UserRoles
            .Where(ur => ur.UserId == currentUserId)
            .Join(_context.Roles, ur => ur.RoleId, r => r.Id, (ur, r) => r.Name)
            .ToListAsync();
    }
}

```

```

if (roles.Contains("CUSTOMER"))
{
    var cskhUser = users.FirstOrDefault(u => u.UserName.Contains("CSKH"));
    selectedUserId = cskhUser?.Id;
}
else if (roles.Contains("CSKH"))
{
    selectedUserId = users.FirstOrDefault()?.Id;
}
}

```

```

ChatViewModel chat = null;
if (!string.IsNullOrEmpty(selectedUserId))
{
    chat = await _messageService.GetMessages(selectedUserId);
    // Đặt ViewData cho CSKH
    if (User.IsInRole("CSKH"))
    {
        var selectedUser = await _context.Users.FindAsync(selectedUserId);
        ViewData["SelectedUserId"] = selectedUserId;
        ViewData["SelectedUserName"] = selectedUser?.UserName ?? "Khách hàng";
    }
}
else
{
    chat = new ChatViewModel
    {
        CurrentUserId = currentUserId,
        ReceiverId = null,
        ReceiverUserName = "Chưa chọn người nhận",
        Messages = new List<UserMessagesListViewModel>()
    };
}

```



```

    }

    var model = new CombinedMessagesViewModel
    {
        Users = users,
        Chat = chat
    };

    return View(model);
}

```

```

public async Task<ActionResult> Chat(string selectedUserId)
{
    var chatViewModel = await _messageService.GetMessages(selectedUserId);
    return View(chatViewModel);
}

```

c. Hệ thống các bảng dữ liệu

