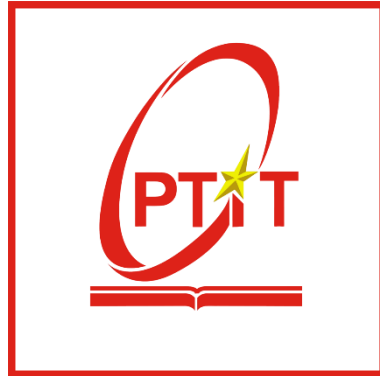


HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG  
KHOA CÔNG NGHỆ THÔNG TIN 1



**BÁO CÁO BÀI TẬP LỚN**  
**BỘ MÔN: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**  
**Đề tài: Quản lý chuỗi cửa hàng quần áo**

Nhóm lớp:	01
Nhóm bài tập lớn:	07
Giảng viên hướng dẫn:	GV. Trần Quốc Khánh
Các thành viên:	B22DCKH004 - Ngô Việt Anh B22DCKH040 - Nguyễn Hải Hiếu B22DCKH044 - Trần Bá Hoàng B22DCKH088 - Lê Đăng Phúc B22DCKH108 - Nguyễn Đình Tiến

Hà Nội, tháng 6, 2025



## MỤC LỤC

<b>I. Mô tả bài toán.....</b>	<b>3</b>
1. Bối cảnh.....	3
2. Mô tả đề tài.....	3
2.1. Nhập dữ liệu.....	3
2.2. ETL (Extract – Transform - Load).....	3
2.3. Báo cáo và trực quan hóa.....	3
2.4. Đối tượng phục vụ.....	3
<b>II. Các chức năng của hệ thống.....</b>	<b>16</b>
1. Sơ đồ chức năng BFD.....	16
2. Mô tả các chức năng.....	16
<b>III. Mô hình quan hệ ERD .....</b>	<b>18</b>
1. Database OutUserDB.....	18
2. Database OnlineDB.....	18
3. Database OfflineDB.....	19
4. Database tập trung DBM.....	20
5. Mô tả các bảng.....	21
<b>IV. Các thủ tục – Procedure .....</b>	<b>25</b>
1. ETL pipeline.....	25
2. Các thủ tục chèn thêm dữ liệu.....	25
3. Các thủ tục ETL.....	31
<b>V. Xây dựng giao diện web.....</b>	<b>38</b>
1. Giao diện đăng nhập, đăng ký.....	38
2. Giao diện trang chủ mua hàng online.....	39
3. Thông tin cá nhân.....	40
4. Trang chi tiết sản phẩm.....	40
5. Trang thanh toán.....	41
6. Trang lịch sử mua hàng.....	42
7. Trang tạo hóa đơn cho khách hàng.....	43
8. Chức năng chấm công cho nhân viên.....	44
9. Trang dashboard.....	44
10. Trang quản lý nhân viên.....	45
11. Xuất báo cáo.....	47
12. Trang quản lý sản phẩm.....	49
<b>VI. Cài đặt .....</b>	<b>52</b>
1. Khởi tạo môi trường.....	52

<b>2. Triển khai hệ thống</b>	<b>52</b>
<b>VII.Đánh giá điểm các thành viên và ký tên.....</b>	<b>53</b>

## **I. Mô tả bài toán**

### **1. Bối cảnh**

Quản lý chuỗi cửa hàng bán quần áo tại Việt Nam, có nhiều cửa hàng tại các thành phố. Với mỗi cửa hàng của hệ thống, các hoạt động đều theo mô hình chung gồm: nhập hàng, bán hàng, tổng chi, doanh thu, lợi nhuận,...

### **2. Mô tả đề tài**

#### **2.1. Nhập dữ liệu**

- Nhập hàng: Nhân viên kiểm tra tồn kho bằng hệ thống, lập danh sách những mặt hàng cần nhập được cung cấp bởi các nhà cung cấp. Khi nhận được hàng, tiến hành kiểm tra chất lượng nếu đúng yêu cầu thì nhân viên tiến hành nhập hàng.
- Bán hàng: Khi khách hàng có nhu cầu mua quần áo, họ sẽ lựa chọn các mẫu có tại cửa hàng. Khi đó, nhân viên xác nhận thông tin cần thiết, sau đó tiến hành tạo biên lai và thanh toán trực tiếp tại quầy cho khách. Khi hóa đơn được xuất, hệ thống sẽ tự động trừ số lượng mặt hàng đã bán. Nhờ đó, cơ sở dữ liệu sẽ quản lý được số lượng còn lại để có kế hoạch nhập hàng hợp lý.
- Doanh thu: Doanh thu là tổng tiền của tiền bán hàng. Doanh thu sẽ được cập nhật theo ngày, tháng, quý, năm.

#### **2.2. ETL (Extract – Transform - Load)**

- Trích xuất dữ liệu từ hệ thống cơ sở dữ liệu của các chi nhánh
- Tiền xử lý dữ liệu, xử lý sai sót và chuẩn hóa thông tin.
- Tải dữ liệu vào một database tập trung để phục vụ thống kê, phân tích.

#### **2.3. Báo cáo và trực quan hóa**

- Hiện thị biểu đồ doanh thu theo thời gian thực giúp người dùng dễ dàng theo dõi.

#### **2.4. Đối tượng phục vụ**

- Quản lý: Theo dõi hiệu suất bán hàng, hàng tồn kho.
- Nhân sự & vận hành: Quản lý nhân viên, tối ưu quy trình hoạt động.

## II. Cơ sở lý thuyết

### 1. Công nghệ Frontend

#### 1.1. HTML (HyperText Markup Language)

- HTML là ngôn ngữ đánh dấu chuẩn để xây dựng bộ khung và nội dung cho mọi trang web. Ngay từ những thẻ cơ bản như `<h1>`–`<h6>` để tạo tiêu đề, `<p>` để định nghĩa đoạn văn, `<img>` để chèn hình ảnh hay `<a>` để thiết lập liên kết, HTML đã đảm bảo rằng trình duyệt có thể “hiểu” và hiển thị đúng cấu trúc thông tin.
- Với sự ra đời của HTML5, khả năng ngữ nghĩa (semantic) được nâng tầm khi xuất hiện các thẻ như `<header>`, `<footer>`, `<nav>`, `<article>`, `<section>` và `<aside>`. Những thẻ này không chỉ giúp mã nguồn trở nên rõ ràng, dễ bảo trì, mà còn cải thiện hiệu quả SEO, hỗ trợ công cụ tìm kiếm định vị nội dung đúng ngữ cảnh và giúp các thiết bị hỗ trợ truy cập (screen readers) phân tích cấu trúc trang một cách chính xác hơn.
- Một trong những điểm đột phá của HTML5 là khả năng nhúng đa phương tiện mà không phụ thuộc plugin bên ngoài. Thẻ `<video>` và `<audio>` cho phép tích hợp video, âm thanh trực tiếp vào trang với các đối tượng điều khiển đơn giản; trong khi `<canvas>` và SVG hỗ trợ vẽ đồ họa động, biểu đồ hoặc hiệu ứng phức tạp ngay trong trình duyệt.
- HTML5 còn mang đến các API lưu trữ cục bộ như Local Storage và Session Storage, giúp các ứng dụng web duy trì trạng thái người dùng, lưu tạm dữ liệu hoặc cache thông tin quan trọng mà không cần tương tác máy chủ. Bên cạnh đó, các thuộc tính cho form (ví dụ `required`, `pattern`, `type="email"`, `type="date"`) giúp thực hiện xác thực ngay trên client, giảm thiểu sai sót và tăng trải nghiệm người dùng.

#### 1.2. CSS (Cascading Style Sheets)

- CSS là ngôn ngữ định kiểu dùng để tách biệt hoàn toàn giữa phần nội dung (HTML) và phần trình bày (giao diện). Nhờ đó, người phát triển có thể kiểm soát mọi khía cạnh về bố cục, màu sắc, font chữ, kích thước, khoảng cách và viền của từng phần tử trên trang web.
- Để xây dựng layout linh hoạt và responsive, CSS cung cấp các mô hình bố cục hiện đại như Flexbox và Grid. Flexbox phù hợp cho các dòng hoặc cột đơn giản, trong khi Grid cho phép tạo các lưới hai chiều phức tạp với ít code hơn.

- Hiệu ứng chuyển động và tương tác trực quan được tạo ra bằng cách sử dụng transition, transform và animation. Những hiệu ứng này không chỉ làm tăng tính thẩm mỹ mà còn cải thiện trải nghiệm người dùng khi duyệt web.
- Các tiền xử lý CSS như Sass và Less mở rộng khả năng viết CSS bằng biến, mixin, nested rules và hàm (functions). Nhờ đó, code gọn gàng hơn, dễ bảo trì và tái sử dụng trong những dự án lớn.
- Frameworks CSS (ví dụ Bootstrap, Tailwind CSS) cung cấp sẵn hệ thống class tiện ích và component có thể dùng ngay lập tức. Việc này giúp rút ngắn thời gian phát triển, đồng thời đảm bảo tính nhất quán về thiết kế và responsive.
- Trong môi trường SPA (Single-Page Application), CSS Modules và Styled Components (trong React/Vue) cho phép cô lập phạm vi style, tránh xung đột tên class và hỗ trợ dynamic styling thông qua JavaScript.
- Việc nắm vững Box Model (margin, border, padding, content), cơ chế cascading và tính specificity của selector là điều then chốt để giải quyết các vấn đề về ưu tiên style và xếp chồng phân lớp một cách chính xác.

### 1.3. JavaScript (JS)

- JavaScript là ngôn ngữ lập trình phía client, cho phép “sống hóa” trang web bằng cách xử lý sự kiện người dùng (click, scroll, keydown...) và tương tác trực tiếp với cấu trúc DOM để thêm, sửa hoặc xóa phần tử. Điều này giúp trang web phản hồi nhanh, không cần tải lại toàn bộ khi cập nhật nội dung.
- Để thực hiện logic tính toán, chuyển đổi dữ liệu và xác thực form ngay trên trình duyệt, JavaScript cung cấp các API mạnh mẽ như Date, Math, JSON và RegExp. Nhà phát triển có thể dễ dàng kiểm tra điều kiện, tính toán phức tạp hoặc xử lý chuỗi để nâng cao chất lượng trải nghiệm.
- Từ phiên bản ES6 trở đi, JavaScript bổ sung cú pháp hiện đại bao gồm let/const để khai báo biến, arrow functions cho biểu thức hàm ngắn gọn, template literals hỗ trợ nội suy chuỗi, destructuring giúp trích xuất giá trị từ object/array, cùng module và class syntax để tổ chức mã nguồn rõ ràng và dễ mở rộng.
- Các công cụ đóng gói module như Webpack, Rollup hay Parcel hỗ trợ gom nhóm và tối ưu hóa file JavaScript thông qua kỹ thuật code splitting và tree shaking, giảm kích thước bundle và tăng tốc độ tải trang.

- Thư viện và framework phổ biến như React, Vue, Angular và Svelte mang đến mô hình component-based, cơ chế quản lý state (Redux, Vuex) và routing cho ứng dụng SPA, giúp tái sử dụng giao diện, duy trì trạng thái và điều hướng liên mạch giữa các “trang” ảo.

#### **1.4. AJAX (Asynchronous JavaScript and XML)**

- AJAX không phải là một thư viện hay ngôn ngữ mới, mà là tập hợp các kỹ thuật cho phép giao tiếp bất đồng bộ giữa client và server mà không cần tải lại toàn bộ trang. Nhờ vậy, ứng dụng web có thể gửi yêu cầu và nhận dữ liệu từng phần, mang lại trải nghiệm mượt mà hơn cho người dùng.
- Về mặt kỹ thuật, AJAX truyền thống sử dụng đối tượng XMLHttpRequest để khởi tạo kết nối, gửi các phương thức GET hoặc POST, sau đó xử lý callback khi server trả về dữ liệu. Mặc dù tên gọi có chữ XML, hầu hết các dự án hiện nay đều ưu tiên định dạng JSON vì nhẹ, dễ parse và tương thích tốt với JavaScript.
- Khi làm việc với tài nguyên chia sẻ giữa các miền khác nhau, CORS (Cross-Origin Resource Sharing) là cơ chế quan trọng để bảo mật. Việc cấu hình header Access-Control-Allow-Origin trên máy chủ quyết định trang nào được phép truy cập tài nguyên, giúp ngăn chặn các request độc hại.
- Nhờ AJAX, các tính năng như gợi ý tìm kiếm tự động (autocomplete), infinite scroll (cuộn tải thêm nội dung), và các thao tác CRUD (tạo, đọc, sửa, xóa) đều có thể thực hiện ngay trong trang hiện tại mà không gây gián đoạn hay mất trạng thái hiển thị trước đó.
- Trong các ứng dụng cần real-time hoặc gần real-time, AJAX thường được kết hợp với WebSocket hoặc Server-Sent Events (SSE) để vừa giữ khả năng bất đồng bộ vừa hỗ trợ đẩy (push) dữ liệu từ server khi có thay đổi, ví dụ trong chat hoặc thông báo cập nhật.
- Việc hiểu rõ cơ chế hoạt động của AJAX, từ luồng request–response đến xử lý lỗi và timeout, giúp nhà phát triển tối ưu hiệu năng, cải thiện trải nghiệm người dùng và giảm thiểu băng thông không cần thiết.



## 2. Công nghệ Backend

### 2.1. Tổng quan về Spring Boot

Spring Boot là một framework mã nguồn mở thuộc hệ sinh thái Spring, được phát triển bởi Pivotal (nay là VMware). Spring Boot được thiết kế để giúp phát triển ứng dụng Java nhanh chóng và đơn giản hơn bằng cách giảm thiểu tối đa cấu hình phức tạp thường gặp trong Spring Framework truyền thống.

#### 2.1.1 Đặc điểm nổi bật

- **Auto-configuration (Tự động cấu hình):** Cơ chế tự động cấu hình của Spring Boot dựa trên việc quét các thư viện phụ thuộc trong classpath để suy luận các cấu hình cần thiết cho ứng dụng. Khi phát hiện một thành phần hoặc thư viện nhất định, Spring Boot sẽ áp dụng một tập hợp cấu hình mặc định phù hợp, bao gồm việc khởi tạo các bean, thiết lập kết nối và cấu hình các thành phần quan trọng mà không yêu cầu lập trình viên phải khai báo chi tiết từng bước. Mô hình này tuân theo nguyên tắc “convention over configuration” nhằm giảm thiểu tác vụ cấu hình thủ công và tránh sai sót, đồng thời cho phép tùy chỉnh khi cần thiết thông qua các thuộc tính cấu hình.
- **Standalone (Ứng dụng độc lập):** Spring Boot hỗ trợ đóng gói ứng dụng dưới dạng tệp thực thi độc lập, bao gồm cả máy chủ ứng dụng nhúng và toàn bộ thư viện cần thiết. Mô hình này giúp ứng dụng có khả năng khởi động tự động mà không phụ thuộc vào môi trường ngoài hay cấu hình máy chủ ứng dụng bên ngoài. Việc triển khai trở nên đơn giản, nhất quán giữa các môi trường, đồng thời giảm thiểu rủi ro do sự khác biệt về cấu hình hoặc phiên bản của máy chủ.
- **Starter Dependencies:** Các starter là các tập hợp thư viện phụ thuộc được Spring Boot định nghĩa sẵn theo nhóm chức năng. Mỗi starter gom lại những thành phần thường dùng cho một mục đích cụ thể và đồng bộ về phiên bản, cho phép lập trình viên chỉ cần khai báo một dependency duy nhất để thiết lập cả một stack chức năng. Cách tiếp cận này đơn giản hóa việc quản lý phụ thuộc, giảm thiểu xung đột phiên bản và đảm bảo tính tương thích giữa các thành phần trong dự án.
- **Actuator:** Actuator là module cung cấp các điểm đầu (endpoint) giám sát và quản lý ứng dụng ở cấp runtime. Thông qua các endpoint này, người vận hành có thể thu thập thông tin về trạng thái hệ thống, các chỉ số hiệu suất, log, cấu hình hiện tại và trạng thái các bean. Actuator hỗ trợ tùy chỉnh mức độ hiển thị, bảo

mật truy cập và tích hợp với các công cụ giám sát bên ngoài, giúp đơn giản hóa việc theo dõi và duy trì độ ổn định của ứng dụng trong môi trường production.

- **Spring Boot CLI:** Spring Boot CLI là công cụ dòng lệnh giúp khởi tạo và chạy nhanh chóng các ứng dụng Spring Boot. CLI cho phép tạo project mẫu với cấu hình mặc định, chạy các script Groovy để thử nghiệm nhanh các thành phần Spring, đồng thời hỗ trợ các lệnh quản lý phụ thuộc và cấu hình. Công cụ này tối ưu cho quá trình phát triển nhanh (rapid prototyping) và thử nghiệm ý tưởng, giúp giảm bớt gánh nặng chuẩn bị môi trường và cài đặt.

### 2.1.2. Ưu điểm khi sử dụng Spring Boot

- **Phát triển nhanh:** Spring Boot loại bỏ hầu hết các bước cấu hình thủ công cho hạ tầng và các thành phần nền tảng thông qua cơ chế auto-configuration và starter dependencies. Nhờ vậy, lập trình viên có thể khởi tạo dự án và triển khai các chức năng nghiệp vụ ngay lập tức mà không phải thiết lập nhiều file cấu hình, giúp rút ngắn đáng kể chu kỳ phát triển và tăng tốc độ đưa sản phẩm ra thị trường.
- **Dễ dàng mở rộng:** Kiến trúc Spring Boot cho phép chia nhỏ ứng dụng thành các module hoặc microservices riêng biệt, mỗi module có thể tự cấu hình và vận hành độc lập. Cơ chế này hỗ trợ tốt cho việc mở rộng quy mô hệ thống, thêm hoặc thay thế các thành phần mà không ảnh hưởng đến toàn bộ kiến trúc, từ đó đáp ứng linh hoạt nhu cầu thay đổi hoặc mở rộng trong tương lai.
- **Tích hợp linh hoạt:** Spring Boot cung cấp các starter tương ứng với nhiều công nghệ phổ biến như JPA/Hibernate, Redis, Kafka, RabbitMQ, Spring Security, v.v. Mỗi starter gom nhóm các thư viện và cấu hình cần thiết để kết nối, giúp lập trình viên dễ dàng tích hợp các thành phần này vào dự án. Điều này đảm bảo tính nhất quán, giảm thiểu xung đột thư viện và đơn giản hóa quá trình cấu hình.
- **Cộng đồng lớn và tài liệu phong phú:** Được phát triển và duy trì bởi Pivotal (hiện là VMware) cùng sự đóng góp của cộng đồng mã nguồn mở, Spring Boot sở hữu một hệ sinh thái rộng lớn với hàng triệu người dùng. Tài liệu chính thức, hướng dẫn, blog kỹ thuật, các khoá học và diễn đàn hỗ trợ sẵn có giúp giải quyết nhanh các thách thức trong quá trình phát triển, đảm bảo việc học tập và tháo gỡ sự cố diễn ra hiệu quả.

### 2.1.3. Ứng dụng của Spring Boot

- **Xây dựng API RESTful:** Spring Boot cung cấp cơ chế tự động cấu hình cho các thành phần liên quan đến giao tiếp HTTP và xử lý JSON, đồng thời tích hợp sẵn các starter hỗ trợ Spring MVC. Qua đó, lập trình viên có thể thiết lập nhanh các controller, cấu hình routing, xử lý request/response và quản lý trạng thái thông qua các annotation cơ bản mà không cần thiết kế bộ khung từ đầu. Cơ chế này đảm bảo tính nhất quán trong việc xây dựng API, đồng thời dễ dàng mở rộng hoặc điều chỉnh theo các tiêu chuẩn REST.
- **Phát triển hệ thống backend cho doanh nghiệp:** Với khả năng tích hợp sâu rộng vào các công nghệ lưu trữ và xử lý dữ liệu (như JPA/Hibernate, transaction management) cùng bộ công cụ bảo mật (Spring Security, OAuth2), Spring Boot là nền tảng lý tưởng để xây dựng các dịch vụ backend phức hợp. Cấu trúc module hóa và khả năng quản lý cấu hình tập trung giúp duy trì tính ổn định, bảo trì và đảm bảo tuân thủ chính sách doanh nghiệp, từ đó đáp ứng các yêu cầu về hiệu suất và bảo mật trong môi trường production.
- **Xây dựng microservices trong kiến trúc phân tán:** Spring Boot kết hợp với Spring Cloud cho phép triển khai các dịch vụ nhỏ gọn, độc lập, mỗi service có thể tự khởi động với cấu hình riêng và giao tiếp qua các giao thức nhẹ (REST/RPC). Mô hình này hỗ trợ cân bằng tải, khám phá dịch vụ, cấu hình tập trung và resilient design thông qua các module như Eureka, Config Server, Circuit Breaker. Nhờ vậy, các microservice có thể vận hành đồng thời và phối hợp linh hoạt, giúp hệ thống dễ dàng mở rộng, chịu lỗi và triển khai liên tục.

### 2.1.4. Kiến trúc tổng quan

- **Quan hệ với Spring Framework:** Spring Boot xây dựng trên nền tảng Spring Framework, sử dụng các module cốt lõi như Spring Core (IoC/DI), Spring MVC (xử lý request–response), Spring Data (tương tác dữ liệu), cùng các module khác tùy theo nhu cầu. Spring Boot đảm nhiệm việc tự động cấu hình và khởi tạo các thành phần này một cách liền mạch, cho phép phát triển nhanh và giữ nguyên tính nhất quán của Spring Framework.
- **Mô hình MVC (Model–View–Controller):** Trong ứng dụng web, Spring Boot thường áp dụng kiến trúc MVC để tách biệt ba thành phần chính:
  - **Model** chịu trách nhiệm quản lý dữ liệu và logic nghiệp vụ.

- **View** đảm nhận việc trình bày giao diện và tương tác với người dùng.
- **Controller** điều phối luồng dữ liệu giữa Model và View, trực tiếp xử lý các HTTP request và mapping hành động. Kiến trúc này giúp duy trì tính rõ ràng, dễ bảo trì và mở rộng.
- **Tổ chức lớp nghiệp vụ:** Spring Boot khuyến khích phân chia mã nguồn theo các tầng rõ ràng:
  - **Controller layer:** Xử lý và nhận request.
  - **Service layer:** Chứa logic nghiệp vụ, orchestrate các thao tác giữa repository và controller.
  - **Repository layer:** Định nghĩa cơ chế truy xuất dữ liệu lên database thông qua Spring Data hoặc JPA.

Việc tách biệt này giúp quản lý phụ thuộc, dễ dàng viết unit test và đảm bảo nguyên tắc Single Responsibility cho mỗi lớp.

## 2.2. Tổng quan về WebSocket

- WebSocket là một giao thức truyền thông mạng chuẩn được thiết kế để tạo kênh giao tiếp hai chiều (full-duplex) giữa client và server qua một kết nối TCP duy nhất. Khác với giao thức HTTP truyền thống, WebSocket cho phép client và server gửi dữ liệu cho nhau bất cứ lúc nào, mà không cần client phải gửi yêu cầu trước.
- WebSocket được chuẩn hóa bởi IETF (RFC 6455) vào năm 2011 và thường được sử dụng trong các ứng dụng web cần tương tác thời gian thực, như chat trực tuyến, game đa người, ứng dụng tài chính, và các hệ thống giám sát.

### 2.2.1. Đặc điểm nổi bật

- **Kênh kết nối hai chiều (Full-Duplex):** WebSocket thiết lập một kênh giao tiếp hai chiều liên tục, cho phép cả client và server có thể gửi và nhận dữ liệu bất kỳ lúc nào mà không cần khởi tạo lại kết nối. Mỗi khung dữ liệu (frame) truyền qua kênh này có thể mang payload dưới dạng văn bản hoặc nhị phân, giúp đảm bảo khả năng tương tác thời gian thực và giảm thiểu độ trễ so với các cơ chế chỉ cho phép luồng đơn hướng.
- **Kết nối duy trì lâu dài:** Quá trình thiết lập WebSocket bắt đầu bằng một HTTP handshake nâng cấp (upgrade handshake). Sau khi handshake hoàn thành, kênh WebSocket sẽ tồn tại cho đến khi một bên (client hoặc server) gửi frame đóng

kết nối. Việc giữ kết nối mở liên tục giúp loại bỏ overhead của việc thiết lập và đóng kết nối lặp lại, đồng thời hạn chế chi phí TCP handshake, cải thiện hiệu suất cho các ứng dụng yêu cầu trao đổi thông tin liên tục.

- **Giao thức nhẹ:** Sau handshake, các message WebSocket được đóng gói trong các frame với header tối giản, chỉ gồm các trường cơ bản như FIN (kết thúc frame), opcode (kiểu dữ liệu) và payload length. Thiết kế này loại bỏ các HTTP header phức tạp cho mỗi lần trao đổi, giúp tiết kiệm băng thông và giảm thiểu overhead xử lý ở cả hai phía. Cơ chế fragmentation cũng cho phép chia nhỏ message lớn thành nhiều frame nhỏ, phù hợp cho truyền tải đa dạng kích thước dữ liệu.
- **Hỗ trợ đa nền tảng:** WebSocket là một tiêu chuẩn mở do IETF (RFC 6455) và W3C định nghĩa, với API JavaScript chuẩn sẵn trong hầu hết các trình duyệt hiện đại. Về phía server, nhiều ngôn ngữ và framework cung cấp thư viện triển khai WebSocket (Java, .NET, Node.js, Python, v.v.). Điều này đảm bảo tính tương thích cao và cho phép xây dựng các hệ thống phân tán, đa nền tảng nơi các thành phần frontend, backend và các dịch vụ khác có thể giao tiếp qua WebSocket một cách liên lạc.

### 2.2.2. Ưu điểm

- **Giảm độ trễ:** WebSocket sử dụng kết nối TCP/TLS duy trì lâu dài, loại bỏ nhu cầu thực hiện lại quá trình thiết lập kết nối và HTTP handshake cho mỗi lần trao đổi dữ liệu. Việc này dẫn tới độ trễ (latency) rất thấp, bởi thời gian chờ giữa các message gần như được rút xuống mức tối thiểu. Trong các ứng dụng yêu cầu phản hồi nhanh, cơ chế này giúp đạt được hiệu suất cao và trải nghiệm mượt mà cho người dùng.
- **Tiết kiệm băng thông:** Sau khi hoàn thành giai đoạn HTTP Upgrade, các message WebSocket chỉ bao gồm khung (frame) với header tối giản, không còn mang theo toàn bộ HTTP header nặng nề. Điều này làm giảm kích thước thông điệp truyền trên mạng, đặc biệt quan trọng trong các kịch bản dữ liệu tần suất cao. Khi tích lũy qua thời gian, băng thông tiết kiệm được sẽ rất đáng kể, giúp giảm tải hạ tầng mạng và chi phí vận hành.
- **Truyền dữ liệu real-time:** WebSocket hỗ trợ mô hình push-based, cho phép server chủ động gửi dữ liệu ngay khi sự kiện xảy ra, thay vì chờ client liên tục

gửi request như trong polling. Điều này đảm bảo thông tin được cập nhật tức thì đến client, phù hợp với các kịch bản đòi hỏi tính thời gian thực cao như chat, cảnh báo, thống kê thị trường hay điều khiển thiết bị từ xa.

- **Tương tác linh hoạt:** Mô hình full-duplex cho phép cả client và server đều có thể khởi tạo và gửi message bất cứ lúc nào, không bị xếp vào hàng đợi request–response truyền thống. Nhờ đó, các luồng dữ liệu tương tác hai chiều phức tạp hay kịch bản publish/subscribe có thể triển khai một cách trực quan. Lập trình viên dễ dàng xây dựng các chức năng như broadcast, back-pressure hay streaming mà không phải phụ thuộc vào phương thức polling kém hiệu quả.

### 2.2.3. Ứng dụng

- **Chat trực tuyến:** WebSocket cho phép tạo kênh giao tiếp hai chiều giữa client và server, nhờ đó tin nhắn có thể được gửi và nhận tức thì mà không cần tải lại trang. Khi người dùng gửi một tin nhắn, client đẩy message lên server qua kết nối WebSocket đã mở sẵn; server sau đó broadcast message đến các client liên quan ngay lập tức. Cơ chế này đảm bảo độ trễ thấp, không phải thực hiện nhiều yêu cầu HTTP, mang lại trải nghiệm trò chuyện mượt mà và gần như thời gian thực.
- **Game đa người:** Trong các trò chơi trực tuyến nhiều người, việc đồng bộ trạng thái game (vị trí nhân vật, hành động, kết quả) giữa các client là rất quan trọng. WebSocket cung cấp kênh full-duplex liên tục, cho phép server cập nhật các event (như di chuyển, va chạm, trạng thái sức khỏe) đến từng client ngay khi phát sinh, đồng thời client có thể gửi lại lệnh hành động mà không có độ trễ của việc thiết lập kết nối. Nhờ vậy, trải nghiệm game trở nên liên tục và đồng bộ chính xác hơn, đặc biệt với các trò chơi cần phản hồi nhanh.
- **Ứng dụng tài chính:** Các ứng dụng tài chính như dashboard chứng khoán, nền tảng giao dịch ngoại hối yêu cầu cập nhật liên tục và tức thì về biến động giá. WebSocket cho phép server push dữ liệu mới ngay khi giá thay đổi, thay vì client phải gửi polling nhiều lần. Điều này không chỉ giảm băng thông mà còn đảm bảo người dùng luôn nhìn thấy con số giá chính xác và kịp thời để ra quyết định giao dịch, giúp hạn chế rủi ro do độ trễ thông tin.
- **Hệ thống giám sát:** Trong các hệ thống IoT hoặc hạ tầng công nghiệp, việc theo dõi trạng thái thiết bị (như cảm biến, máy móc) và báo cáo sự kiện khẩn cấp (như

quá nhiệt, sự cố) ngay khi phát sinh là rất quan trọng. WebSocket cho phép thiết lập kênh liên tục giữa thiết bị hoặc gateway và server giám sát. Khi có bất thường, thiết bị truyền ngay một frame báo động; dashboard của quản trị viên sẽ nhận và hiển thị thông báo tức thì, giúp nhanh chóng xử lý sự cố.

- **Công cụ cộng tác (Collaborative Tools):** Các ứng dụng như soạn thảo văn bản, bảng tính hay whiteboard trực tuyến đa người dùng cùng lúc cần đảm bảo mọi thay đổi được đồng bộ tức thời giữa các client đang tham gia. WebSocket hỗ trợ gửi từng event sửa đổi (chèn, xóa, di chuyển đối tượng) đến server, server tổng hợp và broadcast đến các client khác. Cơ chế này cho phép nhiều người cùng làm việc trên một tài liệu mà không có xung đột dữ liệu, mang đến trải nghiệm cộng tác mượt mà và liền mạch.

#### 2.2.4. Kiến trúc tổng quan

- **Handshake:** Quá trình handshake bắt đầu khi client gửi một yêu cầu HTTP đặc biệt với header Upgrade: websocket và Connection: Upgrade. Yêu cầu này thông báo cho server rằng client mong muốn nâng cấp kết nối từ giao thức HTTP truyền thống sang WebSocket. Server, sau khi xác thực và chấp thuận, phản hồi với status code 101 Switching Protocols cùng các header tương ứng để hoàn tất việc chuyển đổi giao thức. Khi đó, kênh TCP ban đầu được giữ lại nhưng chuyển sang trạng thái WebSocket, sẵn sàng cho trao đổi dữ liệu hai chiều.
- **Kênh truyền dữ liệu:** Sau khi handshake thành công, một kết nối TCP duy nhất sẽ được duy trì liên tục giữa client và server. Mọi thông điệp được đóng gói dưới dạng frame nhỏ, có thể chứa dữ liệu văn bản (text) hoặc nhị phân (binary). Mỗi frame bao gồm header tối giản (ví dụ: FIN, opcode, payload length) và payload thực tế. Cơ chế fragmentation cho phép chia nhỏ những thông điệp lớn thành nhiều frame liên tiếp, đảm bảo tính chính xác và hiệu suất truyền tải. Kênh này cho phép full-duplex, nghĩa là client và server có thể gửi và nhận frame đồng thời mà không chờ nhau.
- **Đóng kết nối:** Khi một bên (client hoặc server) muốn kết thúc phiên giao tiếp, bên đó gửi một frame đóng kết nối (close frame) chứa mã lý do. Đối tác nhận sẽ phản hồi lại với một close frame tương ứng, sau đó tiến hành đóng kết nối TCP một cách an toàn. Quy trình hai bước này đảm bảo rằng cả hai bên đều biết về

việc kết nối bị đóng, đồng thời cho phép giải phóng tài nguyên và socket mà không để lại kết nối “treo” hoặc bị rò rỉ bộ nhớ.

### **3. Cơ sở dữ liệu SQL Server**

- SQL Server là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) do Microsoft phát triển và duy trì, được ứng dụng rộng rãi trong các tổ chức từ doanh nghiệp nhỏ đến tập đoàn lớn.
- Nó sử dụng ngôn ngữ truy vấn mở rộng Transact-SQL (T-SQL) để định nghĩa, truy vấn và thao tác dữ liệu trong các bảng, hỗ trợ các câu lệnh, thủ tục lưu trữ (stored procedures) và trigger phức tạp.
- SQL Server thường được triển khai trong môi trường doanh nghiệp cho các ứng dụng quản lý ERP, CRM, hệ thống báo cáo (BI) và kho dữ liệu (data warehouse), đảm bảo khả năng xử lý giao dịch (OLTP) và phân tích (OLAP) song song.
- Ưu điểm của SQL Server bao gồm: kiến trúc bảo mật đa lớp (authentication, authorization, encryption), tính năng Always On cho khả năng cao sẵn sàng (high availability), mở rộng (scalability) và bản sao (replication) linh hoạt, cùng bộ công cụ Business Intelligence (SSIS, SSAS, SSRS) tích hợp sẵn.
- SQL Server có khả năng tích hợp sâu với nền tảng .NET và các dịch vụ đám mây Azure, cho phép xây dựng các giải pháp hybrid, triển khai linh hoạt và tận dụng các dịch vụ như Azure SQL Database, Azure Synapse Analytics để mở rộng quy mô và cải thiện hiệu suất.

### **4. SQL Server Agent**

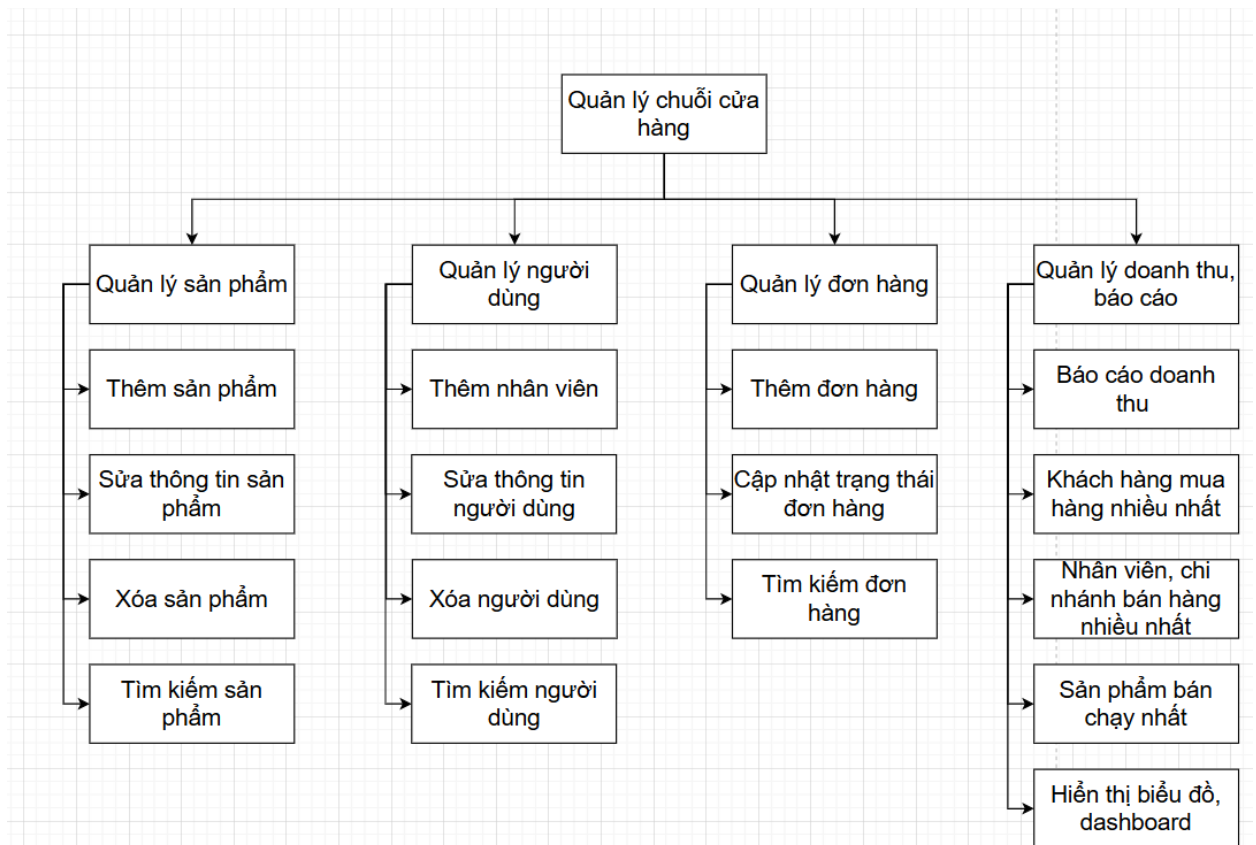
- SQL Server Agent là dịch vụ chạy nền (Windows Service) đi kèm SQL Server, chịu trách nhiệm tự động hoá và lập lịch các tác vụ trong môi trường cơ sở dữ liệu.
- Agent quản lý Jobs, cho phép cấu hình các bước (steps) thực thi tuần tự hoặc song song, bao gồm các lệnh T-SQL, SSIS package, lệnh PowerShell hoặc chương trình bên ngoài.
- Hỗ trợ Schedules, cho phép thiết lập tần suất thực thi (hàng ngày, hàng tuần, hàng tháng hoặc tùy biến theo lịch chi tiết), đảm bảo các tác vụ như sao lưu, tối ưu hoá, index maintenance diễn ra định kỳ.
- Cung cấp cơ chế Alerts và Operators:



- Alerts phát hiện sự kiện hoặc ngưỡng (error severity, performance condition, custom event) và kích hoạt thông báo tự động.
  - Operators định nghĩa người nhận cảnh báo qua email, Net Send hoặc ghi vào Windows Event Log.
- Hỗ trợ Proxies và Credentials, cho phép chạy bước job dưới các tài khoản bảo mật khác nhau, hạn chế quyền truy cập không cần thiết và tuân thủ nguyên tắc least privilege.
  - Tích hợp khả năng Logging và History: ghi lại kết quả thực thi, thời gian chạy, thông báo lỗi giúp theo dõi, phân tích và khắc phục sự cố.
  - SQL Server Agent đảm bảo độ sẵn sàng cao và tính ổn định cho hoạt động tự động hoá, hỗ trợ clustering và Always On Availability Groups, giúp phân phối tải và tăng khả năng chịu lỗi trong môi trường enterprise.

## II. Các chức năng của hệ thống

### 1. Sơ đồ chức năng BFD



### 2. Mô tả các chức năng

#### 2.1. Quản lý sản phẩm:

- Thêm: Bổ sung mặt hàng mới vào hệ thống, tên, size và số lượng của mặt hàng đó
- Xóa: Xóa sản phẩm khỏi hệ thống khi không bán nữa
- Cập nhật: Cập nhật lại tên, giá sản phẩm và các thông tin về sản phẩm đó
- Tìm kiếm: Tìm kiếm theo mã sản phẩm, tên sản phẩm

## **2.2. Quản lý khách hàng, nhân viên:**

- Thêm:
  - Khi có nhân viên mới, admin sẽ tạo cho nhân viên đó một tài khoản để đăng nhập vào hệ thống.
  - Khách hàng tạo tài khoản trên giao diện web.
- Xóa: Khi có nhân viên nghỉ việc, tài khoản nhân viên đó sẽ bị xóa khỏi hệ thống. Đối với khách hàng cho phép chặn khách hàng đăng nhập.
- Cập nhật: chức năng này có thể chỉnh sửa thông tin của nhân viên.
- Tìm kiếm: theo mã hoặc tên của nhân viên, khách hàng.
- Giám sát hiệu suất công việc: Có bảng chấm công cho nhân viên tự động ghi nhận giờ checkin, checkout.

## **2.3. Quản lý đơn hàng:**

- Thêm: Người dùng thêm các mặt hàng vào đơn hàng của mình và thanh toán hoặc mua trực tiếp tại cửa hàng sẽ được nhân viên tạo đơn hàng
- Cập nhật: Cập nhật trạng thái đơn hàng
- Tìm kiếm: Tìm kiếm đơn hàng theo từ khóa tìm kiếm, loại đơn hàng: online hoặc offline và theo khoảng thời gian theo ngày tạo từ ngày đến ngày

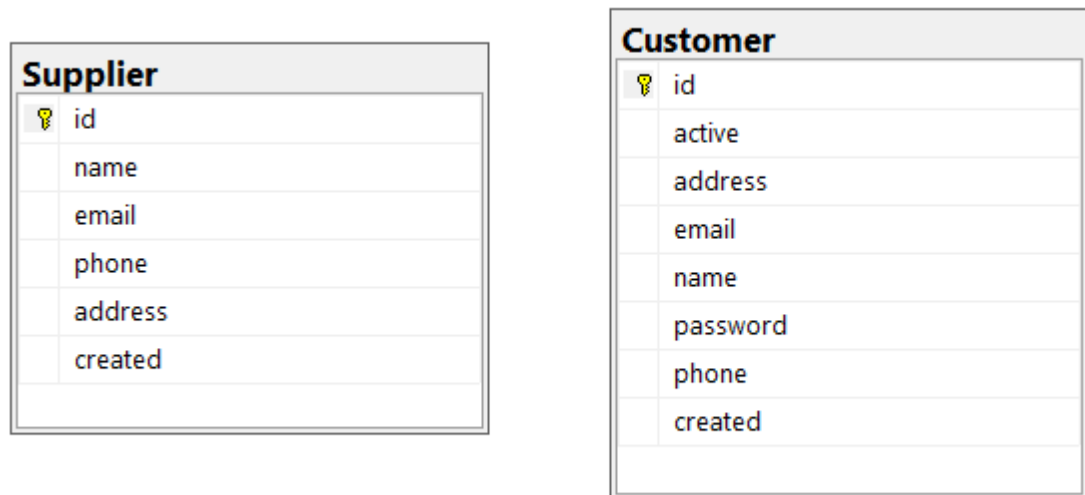
## **2.4. Quản lý doanh thu, báo cáo:**

- Lập báo cáo doanh thu: làm thống kê - báo cáo doanh thu theo thời gian
- Xem khách hàng nào mua hàng online nhiều nhất.
- Xem nhân viên, chi nhánh bán được nhiều hàng nhất.
- Xem sản phẩm bán chạy nhất
- Dashboard thống kê

### III. Mô hình quan hệ ERD

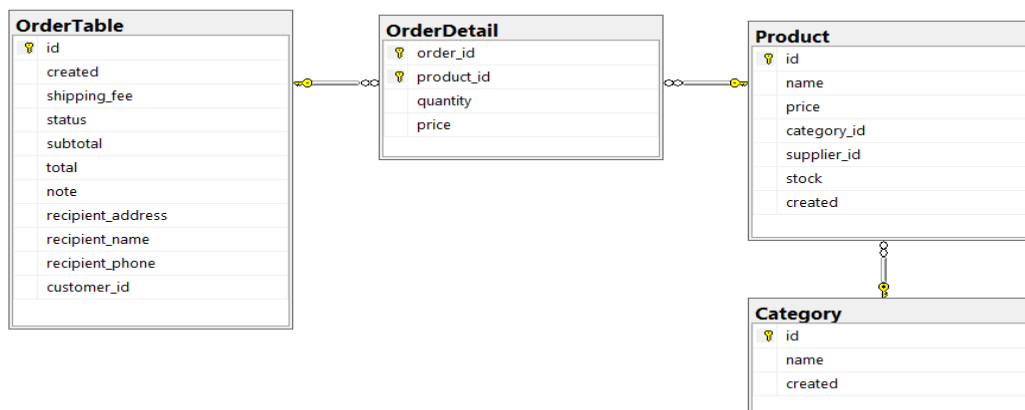
#### 1. Database OutUserDB

Database OutUserDB lưu trữ thông tin của người dùng ngoài như khách hàng và nhà cung cấp sản phẩm.



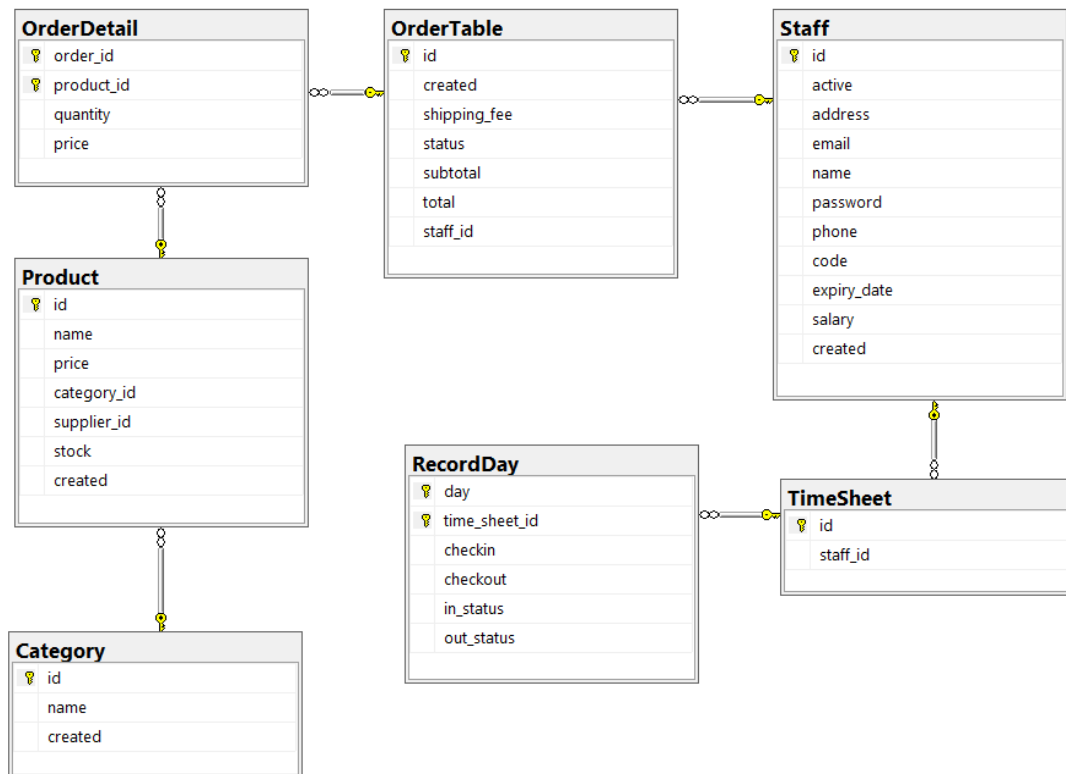
#### 2. Database OnlineDB

Database OnlineDB lưu trữ thông tin các đơn hàng của khách hàng mua sản phẩm online qua trang web của cửa hàng.



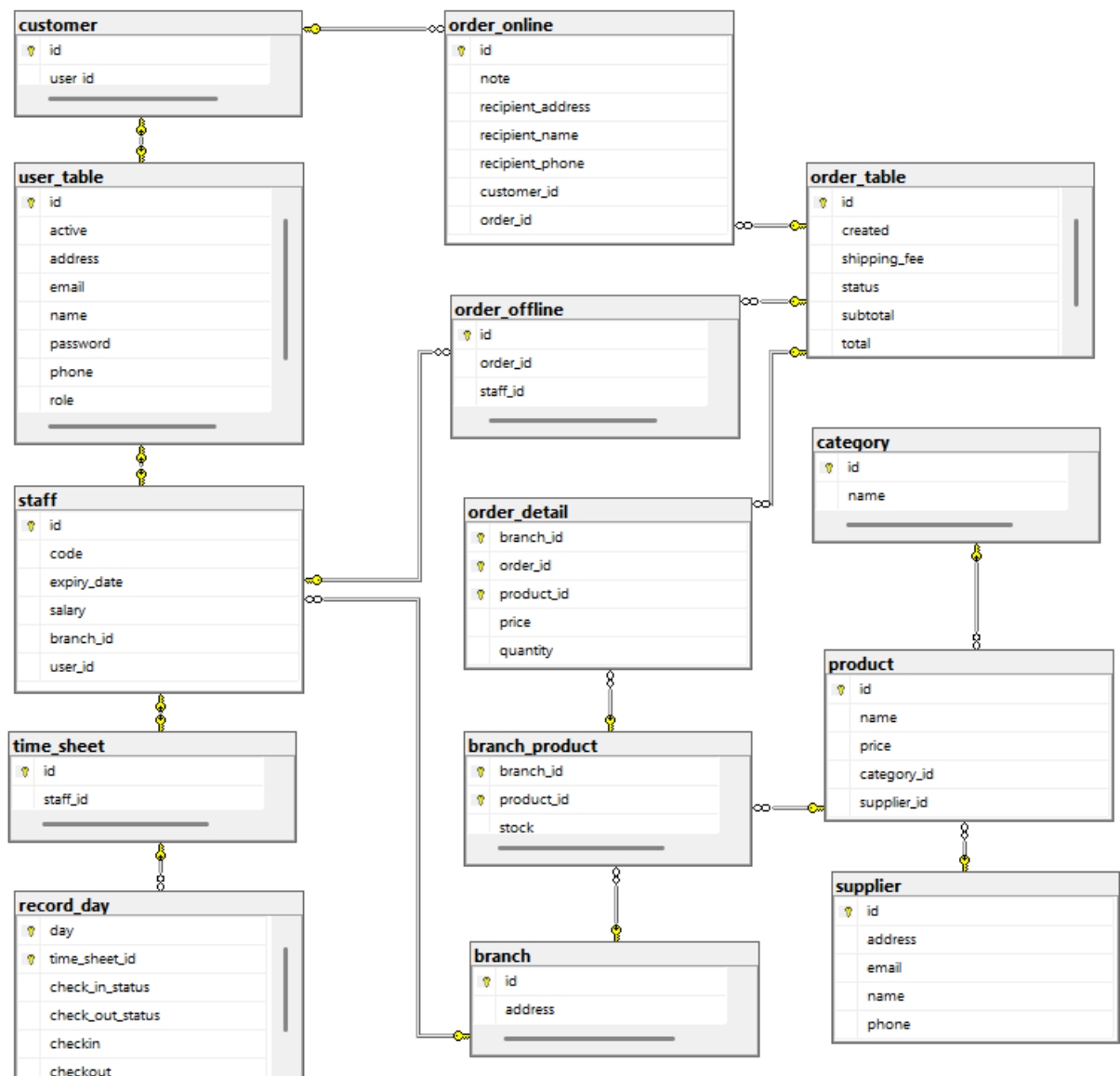
### 3. Database OfflineDB

Database OfflineDB lưu trữ thông tin các đơn hàng của khách hàng mua sản phẩm trực tiếp tại cửa hàng thông qua nhân viên, ngoài ra còn lưu thông tin nhân viên và bảng chấm công cho nhân viên làm việc tại cửa hàng.



#### 4. Database tập trung DBM

Database tập trung DBM sẽ được dùng để tập trung dữ liệu lại một nơi, phục vụ cho việc quản lý, báo cáo và thống kê.



## 5. Mô tả các bảng

### a) dbo.user\_table

Tên trường	Mô tả	Kiểu dữ liệu
Id	Khóa chính	BIGINT
active	Trạng thái hoạt động của người dùng	BIT
address	Địa chỉ của người dùng	VARCHAR(255)
email	Địa chỉ email của người dùng	VARCHAR(255)
name	Tên đầy đủ của người dùng	VARCHAR(255)
password	Mật khẩu người dùng	VARCHAR(255)
phone	Số điện thoại người dùng	VARCHAR(255)
role	Vai trò của người dùng: ADMIN, STAFF, CUSTOMER	VARCHAR(255)

### b) dbo.customer

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
user_id	Khóa ngoại tới bảng user_table	BIGINT

### c) dbo.staff

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
code	Mã nhân viên	VARCHAR(255)
expiry_date	Ngày hết hạn hợp đồng	DATE
salary	Lương tháng của nhân viên	FLOAT
branch_id	Khóa ngoại tới bảng branch	BIGINT
user_id	Khóa ngoại tới bảng user_table	BIGINT

### d) dbo.time\_sheet

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
staff_id	Khóa ngoại tới bảng staff	BIGINT

e) dbo.record\_day

Tên trường	Mô tả	Kiểu dữ liệu
day	Ngày chấm công của nhân viên	DATE
time_sheet_id	Khóa ngoại tới bảng time_sheet	BIGINT
check_in_status	Trạng thái checkin: ONTIME, LATE	VARCHAR(255)
check_out_status	Trạng thái checkout: ONTIME, EARLY	VARCHAR(255)
checkin	Ngày giờ checkin chi tiết	DATETIME2(6)
checkout	Ngày giờ checkout chi tiết	DATETIME2(6)

f) dbo.category

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
name	Tên của loại quần áo	VARCHAR(255)

g) dbo.supplier

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
address	Địa chỉ của nhà cung cấp sản phẩm	VARCHAR(255)
email	Email nhà cung cấp	VARCHAR(255)
name	Tên nhà cung cấp	VARCHAR(255)
phone	Số điện thoại của nhà cung cấp	VARCHAR(255)

h) dbo.branch

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
address	Địa chỉ chi nhánh của cửa hàng	VARCHAR(255)



i) dbo.product

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
name	Tên sản phẩm	VARCHAR(255)
price	Giá sản phẩm	FLOAT
category_id	Khóa ngoại tới bảng category	BIGINT
supplier_id	Khóa ngoại tới bảng supplier	BIGINT

j) dbo.branch\_product

Tên trường	Mô tả	Kiểu dữ liệu
branch_id	Khóa chính và là khóa ngoại tới bảng branch	BIGINT
product_id	Khóa chính và là khóa ngoại tới bảng product	BIGINT
stock	Số lượng tồn kho	BIGINT

k) dbo.order\_table

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
created	Ngày tạo đơn hàng	DATETIME2(6)
shipping_fee	Phí ship	FLOAT
status	Trạng thái đơn hàng	VARCHAR(255)
subtotal	Tiền tính nguyên sản phẩm	FLOAT
total	Tổng tiền	FLOAT

l) dbo.order\_online

Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
note	Ghi chú của khách hàng về đơn hàng	VARCHAR(255)
recipient_address	Địa chỉ nhận hàng	VARCHAR(255)
recipient_name	Tên người nhận hàng	VARCHAR(255)
recipient_phone	Số điện thoại người nhận hàng	VARCHAR(255)
customer_id	Khóa ngoại tới bảng customer	BIGINT
order_id	Khóa ngoại tới bảng order	BIGINT

m) dbo.order\_offline

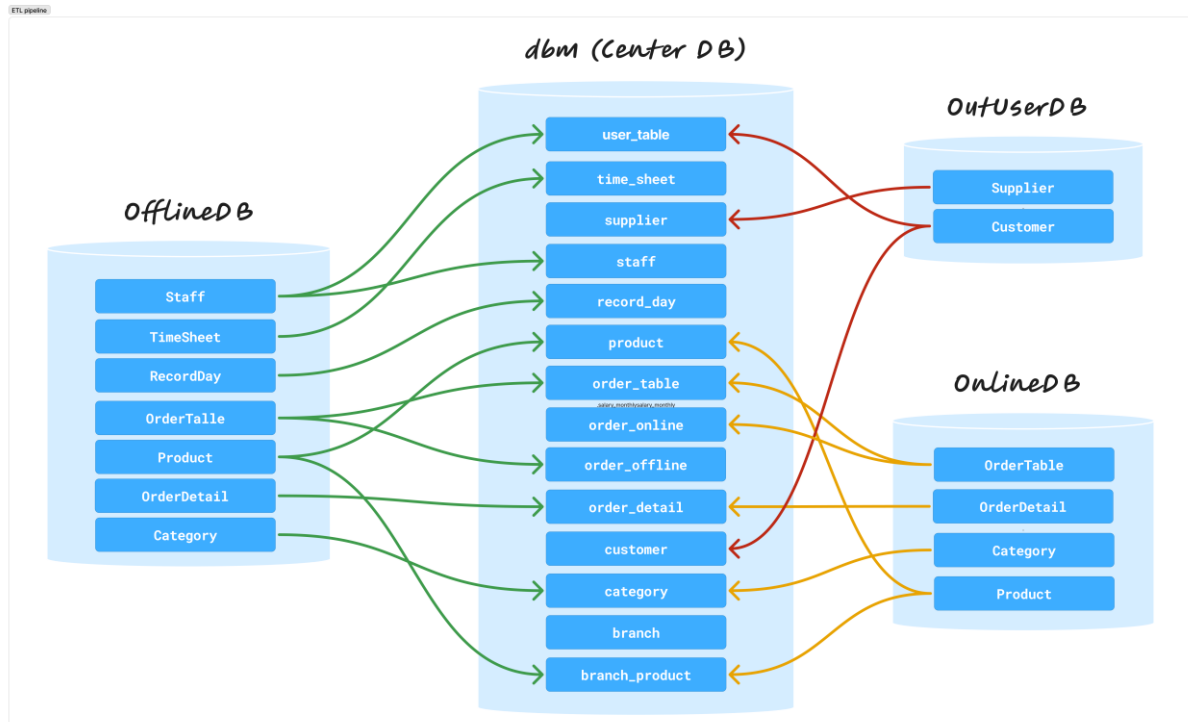
Tên trường	Mô tả	Kiểu dữ liệu
id	Khóa chính	BIGINT
order_id	Khóa ngoại tới bảng order	BIGINT
staff_id	Khóa ngoại tới bảng staff	BIGINT

n) dbo.order\_detail

Tên trường	Mô tả	Kiểu dữ liệu
branch_id	Khóa chính, Khóa ngoại tới bảng branch	BIGINT
order_id	Khóa chính, Khóa ngoại tới bảng order	BIGINT
product_id	Khóa chính, Khóa ngoại tới bảng product	BIGINT
price	Giá của một sản phẩm	FLOAT
quantity	Số lượng của một sản phẩm trong đơn hàng	INT

## IV. Các thủ tục – Procedure

### 1. ETL pipeline



### 2. Các thủ tục chèn thêm dữ liệu

#### a) InsertCustomer

Mô tả:

- Thủ tục InsertCustomer là một công cụ tự động hóa việc thêm khách hàng mới vào hệ thống. Khi được thực thi, nó sẽ tự động xác định số thứ tự tiếp theo cho khách hàng dựa trên số lượng bản ghi hiện có, rồi từ đó xây dựng một địa chỉ email và số điện thoại “giả lập” duy nhất cho khách hàng đó. Ví dụ, nếu hệ thống đã có 9 khách hàng thì thủ tục sẽ gán cho bản ghi mới số thứ tự 10, tạo email dưới dạng email10@gmail.com và số điện thoại 0000000010.
- Tiếp theo, thủ tục sẽ đặt trạng thái khách hàng là “kích hoạt” (active), gán địa chỉ theo mẫu Address 10, tên hiển thị là Customer 10, mật khẩu mặc định là 1234 và lưu lại thời điểm hiện tại làm ngày tạo. Quá trình này hoàn toàn tự động, không cần nhập tay bất kỳ thông tin nào ngoại trừ việc gọi lệnh thực thi.
- Cuối cùng, sau khi ghi bản ghi mới vào bảng, thủ tục sẽ ghi nhận lại mã định danh (ID) mà hệ thống cấp phát cho khách hàng vừa thêm. Mã này có thể được sử dụng ngay lập tức để liên kết với các bảng phụ trợ khác—ví dụ như bảng chi tiết khách hàng, bảng lịch sử giao dịch hoặc bất cứ cấu trúc mở rộng dữ liệu nào

khác—mà không cần phải thực hiện thêm bước tra cứu thủ công. Nhờ vậy, InsertCustomer giúp khởi tạo nhanh chóng các dữ liệu mẫu, phục vụ việc kiểm thử, demo hoặc tạo dữ liệu khởi điểm cho ứng dụng.

b) InsertSupplier

Mô tả:

- Thủ tục InsertSupplier giúp tự động thêm một nhà cung cấp mới vào hệ thống mà không cần nhập liệu thủ công. Khi được gọi, nó sẽ tự động tính ra số thứ tự tiếp theo dựa trên tổng số bản ghi hiện tại trong bảng Supplier, rồi dựa vào số này để sinh ra đầy đủ thông tin: tên nhà cung cấp (Supplier 1, Supplier 2, ...), địa chỉ theo mẫu (Address 1, Address 2,...), email dạng supplier1@gmail.com, supplier2@gmail.com... và số điện thoại (chỉ sử dụng giá trị số thứ tự). Cuối cùng, nó ghi lại thời điểm tạo bản ghi bằng thời gian hiện tại.
- Nhờ cách vận hành này, mỗi lần chạy thủ tục, bạn sẽ có ngay một bản ghi nhà cung cấp mới với dữ liệu mẫu đồng nhất và dễ nhận biết. Điều này vô cùng thuận tiện cho việc khởi tạo dữ liệu thử nghiệm, demo tính năng hoặc tiền xử lý dữ liệu trước khi triển khai thực tế. Bạn cũng có thể mở rộng thủ tục để sau khi chèn xong lấy về mã định danh (ID) của nhà cung cấp rồi sử dụng cho các thao tác liên quan khác, chẳng hạn gán quan hệ sản phẩm–nhà cung cấp hoặc lưu lịch sử giao dịch với nhà cung cấp đó.

c) InsertStaff

Mô tả:

- Thủ tục InsertStaff tự động hóa toàn bộ quá trình thêm một nhân viên mới vào hệ thống, đồng thời khởi tạo luôn thông tin chấm công ban đầu cho người đó. Khi được gọi, nó sẽ:
  - Sinh mã nhân viên theo quy tắc “Staff1”, “Staff2”,... dựa trên số lượng bản ghi hiện có trong bảng Staff.
  - Tạo ngày hết hạn hợp đồng (expiry\_date) ngẫu nhiên trong vòng một năm kể từ ngày chạy, giúp mô phỏng các kỳ hạn khác nhau.
  - Tính mức lương ngẫu nhiên trong khoảng từ 500 đến 2 000, với hai chữ số thập phân, để đa dạng dữ liệu thử nghiệm.
  - Gán trạng thái active mặc định là 1 (đang hoạt động), cùng các thông tin như địa chỉ (Address 1, Address 2,...), email (staff1@example.com,...),

tên hiển thị (Staff 1,...), mật khẩu mặc định là “1234” và số điện thoại giả lập.

- Lưu thời điểm tạo (created) bằng thời gian hiện tại, đảm bảo mọi bản ghi đều có dấu thời gian chính xác.
  - Chèn bản ghi nhân viên mới vào bảng Staff, rồi ngay lập tức lấy lại staff\_id vừa sinh để chèn một bản ghi trống vào bảng TimeSheet, sẵn sàng cho việc cập nhật giờ công sau này.
  - Cuối cùng, in ra thông báo xác nhận hoàn tất việc thêm nhân viên và tạo bảng chấm công.
- Nhờ vậy, chỉ với một lệnh duy nhất, bạn có thể tạo nhanh một hồ sơ nhân viên hoàn chỉnh kèm bản ghi chấm công khởi đầu—rất tiện cho việc khởi tạo dữ liệu demo, thử nghiệm quy trình hoặc tiền xử lý dữ liệu trước khi đưa vào môi trường thực tế.

#### d) InsertCategory

Mô tả:

- Thủ tục InsertCategory giúp bạn đồng bộ việc tạo danh mục (category) mới trên hai hệ thống cơ sở dữ liệu chỉ với một lệnh duy nhất. Khi được gọi, nó sẽ tự động xác định số thứ tự tiếp theo dựa trên tổng số danh mục hiện có trong OnlineDB (ví dụ nếu đã có 3 mục thì mục mới sẽ mang chỉ số 4), sau đó tạo tên danh mục theo mẫu “Category 4”
- Tiếp đó, nó sẽ chèn bản ghi mới vào bảng Category của OnlineDB, gán cho trường created ngày giờ hiện tại để đánh dấu thời điểm tạo. Ngay sau đó, thủ tục tự động nhét thêm cùng một bản ghi danh mục vào bảng Category của OfflineDB, bảo đảm hai cơ sở dữ liệu luôn nhất quán về danh sách danh mục. Nhờ đó, bạn không cần phải thực hiện hai thao tác riêng biệt, giảm thiểu sai sót và đơn giản hóa quy trình quản lý dữ liệu.

e) InsertProduct

Mô tả:

- Thủ tục InsertProduct là một giải pháp tự động hóa việc thêm sản phẩm mới vào cả hai cơ sở dữ liệu trực tuyến và ngoại tuyến, đảm bảo tính nhất quán và đa dạng dữ liệu khi khởi tạo.
- Khi bạn thực thi EXEC InsertProduct;, đầu tiên nó sẽ chuyển ngữ cảnh sang database OnlineDB. Trong phạm vi thủ tục, các biến tạm được khai báo để lưu trữ tên sản phẩm, giá, khoá ngoại liên kết với danh mục và nhà cung cấp, cùng số lượng tồn kho tương ứng cho cả hai hệ thống.
- Đầu tiên, thủ tục xác định tên sản phẩm bằng cách đếm số bản ghi hiện có trong bảng Product của OnlineDB và cộng thêm 1. Kết quả là chuỗi “Product X” với X là số thứ tự tiếp theo. Tiếp đó, nó tạo ra giá bán ngẫu nhiên trong khoảng từ 0 đến 1.000, làm tròn đến hai chữ số thập phân, giúp mô phỏng đa dạng mức giá sản phẩm.
- Để gán quan hệ với danh mục, thủ tục dùng câu lệnh SELECT TOP 1 ... ORDER BY NEWID() để chọn ngẫu nhiên một category\_id từ bảng Category của OnlineDB. Tương tự, một supplier\_id ngẫu nhiên được rút từ bảng Supplier trong cơ sở dữ liệu OutUserDB, giúp sản phẩm vừa chèn có mối liên kết tự nhiên với nhà cung cấp.
- Về tồn kho, hai biến @stock\_db1 và @stock\_db2 lần lượt được gán giá trị ngẫu nhiên — mỗi giá trị nằm trong khoảng 0 đến 6.000.000, làm tròn đến bội số 5.000.000. Mục đích là giả lập số lượng hàng hoá tồn kho khác nhau giữa hệ thống online và offline.
- Cuối cùng, thủ tục thực hiện hai lần chèn (INSERT): lần đầu vào bảng OnlineDB.dbo.Product, và ngay sau đó một bản sao gần như giống hệt được chèn vào bảng OfflineDB.dbo.Product. Các trường được điền bao gồm tên, giá, category\_id, supplier\_id, số lượng tồn kho tương ứng và ngày giờ tạo (dùng GETDATE()).
- Nhờ vậy, chỉ với một lệnh gọi duy nhất, bạn sẽ có một sản phẩm mới tự động sinh dữ liệu mô phỏng, đã liên kết với danh mục và nhà cung cấp, đồng thời đồng bộ tồn kho giữa hai cơ sở dữ liệu, rất tiện cho việc khởi tạo dữ liệu thử nghiệm hoặc đồng bộ hóa môi trường phát triển và vận hành.

f) InsertOfflineOrder

Mô tả:

- Khi bạn gọi EXEC InsertOfflineOrder, hệ thống sẽ tự động tạo một đơn hàng “offline” hoàn chỉnh mà không cần nhập tay từng bước. Đầu tiên, thủ tục ghi nhận thời điểm hiện tại vào biến @created, sau đó chọn ngẫu nhiên trạng thái đơn (CANCELLED, COMPLETED hoặc PENDING) và một nhân viên phụ trách từ bảng Staff. Phí vận chuyển được gán mặc định, còn số lượng mặt hàng trong đơn được sinh ngẫu nhiên từ 1 đến 3, rồi biến @subtotal và @total được khởi tạo tương ứng.
- Tiếp theo, thủ tục chèn bản ghi chính vào bảng OrderTable với các trường ngày tạo, phí vận chuyển, trạng thái, tổng phụ, tổng cộng và nhân viên. Ngay sau khi chèn, nó gọi SCOPE\_IDENTITY() để lấy mã đơn hàng vừa sinh. Thủ tục tiếp tục chọn ngẫu nhiên mỗi sản phẩm (từ 1 đến 3 món) không trùng lặp từ bảng Product, lưu tạm vào một bảng tạm, rồi mở con trỏ duyệt qua danh sách này.
- Với mỗi sản phẩm trong con trỏ, thủ tục sinh số lượng mua ngẫu nhiên từ 1 đến 5, kiểm tra tồn kho hiện tại và dừng lại nếu số lượng yêu cầu vượt quá hàng có sẵn. Nếu vẫn còn đủ hàng, nó tính thành tiền cho dòng đó, cộng dồn vào tổng phụ, ghi chi tiết vào OrderDetail và giảm tồn kho tương ứng trong bảng Product. Khi đã xử lý xong tất cả sản phẩm, thủ tục đóng con trỏ, xóa bảng tạm, rồi tính lại tổng đơn hàng (tổng phụ cộng với phí vận chuyển) và cập nhật kết quả cuối cùng vào OrderTable. Nhờ vậy, chỉ với một lệnh gọi duy nhất, bạn có thể mô phỏng hoặc khởi tạo dữ liệu đơn hàng offline đầy đủ và chính xác.

g) InsertOnlineOrder

Mô tả:

- Thủ tục InsertOnlineOrder tự động hóa toàn bộ quy trình tạo một đơn hàng trực tuyến mới, từ khi bắt đầu khởi tạo đến khi hoàn tất tính toán tổng tiền. Đầu tiên, nó ghi lại thời điểm hiện tại và sinh ngẫu nhiên trạng thái đơn hàng (CANCELLED, COMPLETED hoặc PENDING), đồng thời tạo một ghi chú duy nhất dưới dạng UUID. Tiếp đến, thủ tục tự động chọn một khách hàng ngẫu nhiên từ bảng Customer của cơ sở dữ liệu OutUserDB, lấy về địa chỉ, tên, số điện thoại và mã khách hàng; nếu bất cứ thông tin bắt buộc nào bị thiếu, thủ tục sẽ báo lỗi và dừng lại.

- Sau đó, nó sinh ngẫu nhiên phí vận chuyển trong khoảng 5–50 đơn vị tiền tệ và xác định số lượng sản phẩm trong đơn (từ 1 đến 3 mặt hàng). Với các giá trị này, thủ tục chèn một bản ghi vào OrderTable, lấy về mã đơn hàng vừa tạo. Sau khi có mã đơn, nó tiếp tục chọn ngẫu nhiên từng sản phẩm từ bảng Product trong OnlineDB, đảm bảo không trùng lặp, và lưu tạm vào một bảng tạm.
- Khi xử lý chi tiết đơn, thủ tục duyệt tuần tự mỗi sản phẩm đã chọn, sinh số lượng mua ngẫu nhiên (1–5), kiểm tra tồn kho, và nếu số lượng yêu cầu vượt quá hàng có sẵn thì sẽ báo lỗi và dừng. Nếu đủ hàng, nó tính thành tiền cho mỗi dòng, cộng dồn vào tổng phụ, chèn dòng đó vào OrderDetail và đồng thời giảm tồn kho tương ứng trong bảng Product. Khi đã xử lý hết các sản phẩm, thủ tục tính toán lại tổng đơn (bao gồm phí vận chuyển và tổng phụ) rồi cập nhật lại OrderTable để hoàn thiện thông tin thanh toán.
- Chỉ với một lệnh gọi duy nhất, InsertOnlineOrder sẽ tạo ra một đơn hàng hoàn chỉnh: từ lựa chọn khách hàng, sinh phí vận chuyển, chọn và kiểm soát tồn kho, đến ghi chi tiết sản phẩm và cập nhật tổng tiền, rất thuận tiện cho việc khởi tạo dữ liệu thử nghiệm hoặc mô phỏng quy trình bán hàng thực tế.

#### h) InsertRecordDay

Mô tả:

- Thủ tục InsertRecordDay tự động hóa quy trình ghi nhận chấm công hàng ngày cho một nhân viên bất kỳ trong hệ thống. Khi được gọi, nó sẽ:
  1. Xác định ngày chấm công là ngày hiện tại.
  2. Chọn ngẫu nhiên một nhân viên từ danh sách trong bảng nhân viên.
  3. Tìm khóa chấm công (time sheet) tương ứng với nhân viên đó.
  4. Kiểm tra xem trong ngày hôm nay đã có bản ghi chấm công cho nhân viên này chưa; nếu đã có, thủ tục sẽ dừng ngay và thông báo lỗi để tránh trùng lặp.
  5. Sinh ra giờ vào làm việc một cách ngẫu nhiên trong khoảng từ 7:30 đến 8:30 sáng, rồi dựa vào kết quả đó xác định xem nhân viên có đi muộn (Late) hay đúng giờ (OnTime).
  6. Sinh ra giờ ra (checkout) trong khoảng từ 4:30 đến 5 giờ chiều, rồi đánh dấu trạng thái về sớm (Early) hoặc đúng giờ (OnTime) tùy thuộc vào thời gian thực tế.



7. Cuối cùng, lưu lại dữ liệu chấm công — bao gồm ngày, mã chấm công, giờ vào, giờ ra, trạng thái vào và trạng thái ra — vào bảng lưu trữ.
  - Nhờ vậy, chỉ với một lệnh duy nhất, hệ thống sẽ tự động tạo một bản ghi chấm công hoàn chỉnh, từ việc chọn nhân viên, kiểm soát trùng lặp, sinh giờ công hợp lệ đến đánh dấu trạng thái, rất thuận tiện cho việc khởi tạo dữ liệu demo hoặc mô phỏng luồng làm việc thực tế.
- i) SetStock
- Mô tả:
- Cập nhật số lượng sản phẩm

### 3. Các thủ tục ETL

- a) EtlCustomer
- Mô tả:
- Thủ tục EtlCustomer tự động hóa toàn bộ quy trình đồng bộ dữ liệu khách hàng từ bảng Customer trong cơ sở dữ liệu OutUserDB vào hai bảng user\_table và customer trong cơ sở dữ liệu DBM. Khi bạn gọi EXEC EtlCustomer;, nó sẽ:
    1. Chạy lệnh MERGE giữa bảng đích user\_table và bảng nguồn OutUserDB.dbo.Customer, dùng email làm khóa đối chiếu để tránh trùng lặp.
    2. Với những bản ghi đã có sẵn (matched), nếu bất kỳ thông tin nào—như trạng thái kích hoạt, địa chỉ, tên, số điện thoại, mật khẩu—khác so với bản gốc, thủ tục sẽ cập nhật lại tất cả các trường đó đồng thời gán thêm vai trò ‘CUSTOMER’.
    3. Với những khách hàng mới (not matched by target), thủ tục sẽ chèn toàn bộ thông tin vào user\_table (active, address, email, name, phone, password, role = ‘CUSTOMER’) và đồng thời ghi lại tất cả các user\_id vừa được tạo vào một bảng tạm nội bộ.
    4. Sau khi MERGE xong, thủ tục sẽ lấy danh sách user\_id mới từ bảng tạm và chèn vào bảng customer trong DBM, đảm bảo mỗi tài khoản user đều có một bản ghi tương ứng trong bảng chi tiết khách hàng.
    5. Cuối cùng, thủ tục in ra thông báo xác nhận hoàn thành quá trình ETL.
  - Nhờ vậy, chỉ với một lệnh gọi duy nhất, toàn bộ khách hàng từ hệ thống nguồn sẽ được cập nhật hoặc bổ sung sang hệ thống DBM một cách nhất quán và không

trùng lặp, đồng thời phân tách rõ ràng giữa thông tin tài khoản (user\_table) và thông tin đặc thù khách hàng (customer).

b) EtlSupplier

Mô tả:

- Thủ tục EtlSupplier giúp bạn đồng bộ tự động toàn bộ danh sách nhà cung cấp giữa hệ thống nguồn và hệ thống DBM chỉ với một lệnh duy nhất. Khi chạy EXEC EtlSupplier;, đầu tiên nó sẽ chuyển ngữ cảnh về cơ sở dữ liệu DBM, rồi thực hiện một phép MERGE giữa bảng đích supplier và bảng nguồn OutUserDB.dbo.Supplier, dùng ID nhà cung cấp để đối chiếu.
- Nếu một nhà cung cấp đã tồn tại ở cả hai bên nhưng có sự khác biệt về tên, email, điện thoại hoặc địa chỉ, thủ tục sẽ tự động cập nhật lại các trường đó trong bảng đích cho khớp với dữ liệu nguồn. Với những nhà cung cấp mới chỉ có trong hệ thống nguồn, nó sẽ chèn thêm vào bảng đích đầy đủ thông tin (tên, email, điện thoại, địa chỉ), đảm bảo không bỏ sót.
- Ngược lại, nếu một bản ghi chỉ còn xuất hiện ở bảng đích mà không có trong nguồn nữa, thủ tục sẽ xóa bỏ bản ghi đó để giữ cho dữ liệu luôn đồng nhất. Cuối cùng, sau khi hoàn tất, nó sẽ in ra thông báo xác nhận quá trình đồng bộ đã thành công, giúp bạn yên tâm rằng danh sách nhà cung cấp trong DBM luôn chính xác và cập nhật kịp thời mà không phải thao tác thủ công.

c) EtlStaff

Mô tả:

- Thủ tục EtlStaff tự động hóa quy trình đồng bộ thông tin nhân viên từ hệ thống nguồn sang hệ thống DBM, bao gồm hai phần chính: cập nhật bảng tài khoản người dùng và chèn cập nhật bảng nhân viên chi tiết.
- Khi bạn gọi EXEC EtlStaff;, đầu tiên thủ tục sẽ so sánh và hợp nhất (MERGE) dữ liệu giữa bảng OfflineDB.dbo.Staff (nguồn) và bảng DBM.dbo.user\_table (đích), dùng email để tránh trùng lặp.
  1. Với các bản ghi đã tồn tại: nếu bất kỳ thông tin nào—như trạng thái kích hoạt, địa chỉ, tên, số điện thoại hay mật khẩu—khác biệt so với dữ liệu nguồn, thủ tục sẽ cập nhật lại cho khớp và gán thêm vai trò ‘STAFF’.
  2. Với những tài khoản người dùng mới: nó sẽ chèn bản ghi mới vào user\_table (bao gồm trạng thái, địa chỉ, email, tên, điện thoại, mật khẩu và

role = 'STAFF'), đồng thời ghi lại ID người dùng vừa tạo cùng thông tin code, expiry\_date và salary vào một bảng tạm.

- Sau khi hoàn thành phần MERGE, thủ tục sẽ kiểm tra bảng tạm: nếu có các bản ghi mới, nó sẽ chèn các dòng này vào bảng DBM.dbo.staff, đảm bảo mỗi tài khoản người dùng ứng với một hồ sơ nhân viên chi tiết với đầy đủ mã nhân viên (code), ngày hết hạn hợp đồng và mức lương.
- Tiếp theo, để đảm bảo dữ liệu đầy đủ, thủ tục sẽ duyệt bảng DBM.dbo.staff và gán cho mỗi nhân viên có branch\_id còn trống giá trị mặc định là 2.
- Cuối cùng, một thông báo được in ra xác nhận quá trình ETL đã hoàn tất thành công, giúp bạn yên tâm rằng hệ thống DBM đã được cập nhật đầy đủ, nhất quán và không bỏ sót bất cứ nhân viên nào.

#### d) EtlProduct

Mô tả:

- Thủ tục EtlProduct tự động hóa toàn bộ quy trình thu thập, làm sạch và đồng bộ thông tin danh mục, nhà cung cấp, sản phẩm và tồn kho giữa ba hệ thống nguồn (OfflineDB, OnlineDB và OutUserDB) vào cơ sở dữ liệu đích DBM. Khi bạn gọi EXEC EtlProduct;, nó sẽ lần lượt thực hiện các bước sau trong một luồng ETL hoàn chỉnh:
  1. Đồng bộ danh mục (Category): Trước hết, thủ tục gom lại toàn bộ tên danh mục từ cả hai hệ thống OfflineDB và OnlineDB, sau đó gộp thành một tập duy nhất – loại bỏ trùng lặp thông qua phép UNION. Với mỗi tên danh mục mới mà DBM chưa có, nó tự động chèn thêm vào bảng dbm.dbo.category. Nhờ vậy, bảng danh mục ở DBM luôn mở rộng kịp thời mọi hạng mục sản phẩm mới phát sinh ở bất kỳ nguồn nào.
  2. Chuẩn bị dữ liệu nhà cung cấp (Supplier): Đoạn mã liên quan đến nhà cung cấp tạm thời được chú thích (comment) vì cần bật tắt tính năng IDENTITY\_INSERT và đảm bảo chỉ nhập những nhà cung cấp thực sự liên quan đến sản phẩm. Về cơ bản, ý tưởng là chọn ra tất cả những nhà cung cấp đang có mặt trong bất kỳ sản phẩm nào của OfflineDB hoặc OnlineDB, rồi chèn hoặc cập nhật chúng vào bảng dbm.dbo.supplier. Cách làm này đảm bảo chỉ đồng bộ những nhà cung cấp thực sự cần thiết, tránh kéo vào các bản ghi không liên quan.

3. Đồng bộ sản phẩm (Product): Đầu tiên, thủ tục khớp mỗi sản phẩm nguồn với danh mục tương ứng trên DBM thông qua một bảng tạm CategoryMapping, nơi lưu cặp source\_id → target\_id. Tiếp theo, nó gom tất cả sản phẩm từ hai nguồn OfflineDB và OnlineDB, gắn thêm thông tin “Offline” hoặc “Online”, rồi dùng hàm ROW\_NUMBER để ưu tiên bản ghi Online nếu cùng một tên sản phẩm xuất hiện ở cả hai nơi. Với danh sách sản phẩm đã làm sạch này, thủ tục thực hiện phép MERGE vào bảng dbm.dbo.product: nếu DBM đã có sản phẩm cùng tên, nó cập nhật lại danh mục, nhà cung cấp và giá; nếu chưa có, nó chèn thêm bản mới. Nhờ chiến thuật chọn lọc ưu tiên bản Online, DBM luôn duy trì mức giá và thông tin sản phẩm chính xác nhất.
  4. Đồng bộ tồn kho chi nhánh (Branch\_Product): Cuối cùng, thủ tục gom tồn kho của mỗi sản phẩm ở hai chi nhánh: chi nhánh online (branch\_id = 1) và offline (branch\_id = 2). Nó lấy số liệu tồn kho từ bảng Product tương ứng, gộp vào chung một tập rồi chọn duy nhất một bản ghi cho mỗi cặp (chi nhánh, tên sản phẩm). Sau đó, phép MERGE sẽ cập nhật bảng dbm.dbo.branch\_product: tăng, giảm hoặc thêm mới các bản ghi sao cho số liệu tồn kho trên DBM luôn khớp với hai hệ thống nguồn.
- Khi hoàn thành, thủ tục in ra dòng thông báo “ETL process completed successfully.” cho biết toàn bộ quá trình đồng bộ đã diễn ra thành công. Nhờ EtlProduct, bạn có thể giữ cho DBM luôn cập nhật đầy đủ – từ danh mục, nhà cung cấp, sản phẩm đến tồn kho chi nhánh – chỉ với một lệnh duy nhất, không cần can thiệp thủ công từng bước.
- e) EtlOfflineOrder
- Mô tả:
- Thủ tục EtlOfflineOrder bắt đầu bằng việc mở một giao dịch lớn để đảm bảo rằng toàn bộ quá trình chuyển đơn hàng offline diễn ra an toàn và nhất quán. Ngay khi chạy, nó sẽ báo cho bạn biết có bao nhiêu đơn hàng đang nằm trong hai bảng đích (order\_table và order\_offline), giúp bạn nắm rõ khối lượng dữ liệu trước khi di chuyển. Tiếp theo, thủ tục tạo một bảng tạm chứa tất cả các đơn hàng từ hệ thống OfflineDB mà chưa được ghi nhận vào order\_offline; nếu không tìm thấy đơn

hàng mới nào, nó sẽ dọn dẹp bảng tạm, cam kết giao dịch và kết thúc ngay, giúp tránh lãng phí tài nguyên.

- Nếu có đơn mới, EtlOfflineOrder sẽ liệt kê danh sách ID để bạn dễ theo dõi rồi dùng con trỏ để lặp qua từng đơn. Với mỗi đơn, nó chèn một bản ghi tương ứng vào order\_table của DBM, ngay lập tức lấy về ID mới và lưu lại mối quan hệ giữa ID cũ và ID mới vào một bảng tạm thứ hai. Tiếp đến, thủ tục bật IDENTITY\_INSERT để ghi bản ghi trong order\_offline, sử dụng chính ID gốc làm khóa chính và ID mới làm tham chiếu, đồng thời ghi kèm mã nhân viên chịu trách nhiệm.
- Sau khi chuyển xong thông tin đơn hàng chính, nó chuyển sang xử lý chi tiết: tạo bảng tạm cho chi tiết từng sản phẩm, kéo dữ liệu từ OfflineDB.dbo.OrderDetail (qua bảng ánh xạ ID), gán cho mỗi dòng chi tiết nhánh offline, rồi chèn vào order\_detail sử dụng ID mới của đơn. Khi đã hoàn thành, tất cả bảng tạm đều được xóa sạch, và EtlOfflineOrder in ra tổng số bản ghi cuối cùng trong ba bảng đích để bạn kiểm chứng.
- Cuối cùng, nếu toàn bộ các bước trên không phát sinh lỗi, thủ tục sẽ cam kết giao dịch và thông báo “ETL process completed successfully for Offline Orders.”; ngược lại, nếu có bất cứ lỗi nào xảy ra trong quá trình, nó sẽ lập tức hoàn tác toàn bộ, đồng thời in đầy đủ thông báo lỗi—từ dòng mã vấp phải đến mức độ nghiêm trọng—giúp bạn dễ dàng tìm và sửa vấn đề. Nhờ vậy, quá trình di cư đơn hàng offline luôn diễn ra trơn tru, an toàn và có thể theo dõi chi tiết.

f) EtlOnlineOrder

Mô tả:

- Khi bạn chạy thủ tục EtlOnlineOrder, đầu tiên nó sẽ mở một giao dịch lớn để đảm bảo rằng mọi thay đổi đều được thực hiện đồng bộ và có thể hoàn tác nếu gặp lỗi. Ngay lập tức, thủ tục in ra số lượng đơn đang tồn tại trong hai bảng đích (order\_table và order\_online), giúp bạn nắm được tình trạng hiện tại trước khi di chuyển dữ liệu. Tiếp đó, nó tạo một bảng tạm chứa tất cả các đơn hàng từ OnlineDB.dbo.OrderTable mà chưa có trong DBM.dbo.order\_online. Nếu không có đơn nào cần di chuyển, thủ tục sẽ xóa bảng tạm, cam kết giao dịch và kết thúc sớm, tiết kiệm tài nguyên.

- Trong trường hợp có đơn mới, thủ tục sẽ liệt kê danh sách ID để bạn kiểm tra, rồi sử dụng một con trỏ để duyệt tuần tự từng đơn trong bảng tạm. Với mỗi đơn, nó chèn dữ liệu ngày tạo, phí vận chuyển, trạng thái và các tổng phụ vào bảng chung `order_table`, sau đó ngay lập tức ghi nhận ID mới sinh ra và lưu lại ánh xạ giữa ID gốc và ID mới. Tiếp theo, thủ tục bật tính năng `IDENTITY_INSERT` để tận dụng chính ID gốc của đơn hàng làm khóa chính cho bảng `order_online`. Mỗi bản ghi trong bảng chuyên biệt này sẽ mang theo ID gốc, ID mới, mã khách hàng, thông tin người nhận (tên, số điện thoại, địa chỉ) và ghi chú.
- Sau khi đảm bảo đơn chính đã được lưu đúng, thủ tục chuyển sang xử lý chi tiết từng mặt hàng. Nó tạo một bảng tạm mới, kéo chi tiết đơn từ `OnlineDB.dbo.OrderDetail` rồi nối với bảng ánh xạ ID để xác định ID mới. Sau đó, nó in ra số dòng cần xử lý, tự động gán mỗi dòng một `branch_id` phù hợp (lấy từ bảng `branch_product`), và chèn những dòng có chi nhánh hợp lệ vào bảng `order_detail`, liên kết chính xác với đơn hàng trong `order_table`.
- Cuối cùng, toàn bộ bảng tạm sẽ được xóa sạch, thủ tục in ra số lượng bản ghi cuối cùng trong ba bảng đích để bạn đối chiếu, rồi cam kết giao dịch và thông báo “ETL process completed successfully for Online Orders.” Trong trường hợp bất kỳ lỗi nào xảy ra trong suốt quá trình, mọi thay đổi sẽ được hoàn tác ngay lập tức và thủ tục sẽ in đầy đủ thông tin lỗi—from thông điệp đến dòng mã, tên thủ tục, số lỗi, độ nghiêm trọng và trạng thái—giúp bạn nhanh chóng xác định và xử lý các vấn đề phát sinh. Nhờ vậy, dữ liệu đơn hàng online luôn được di chuyển một cách an toàn, nhất quán và minh bạch.

g) `EtlTimeSheet`

Mô tả:

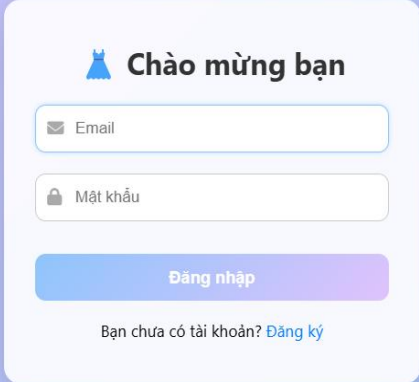
- Thủ tục `EtlTimeSheet` được thiết kế để đồng bộ toàn bộ dữ liệu chấm công từ hệ thống nguồn vào cơ sở dữ liệu DBM một cách trọn vẹn, bao gồm cả khung nhân viên, bảng chấm công hàng ngày và kết quả check-in/check-out. Khi bạn gọi `EXEC EtlTimeSheet`, đầu tiên nó sẽ thực thi thủ tục `EtlStaff`, đảm bảo rằng tất cả nhân viên từ hệ thống `OfflineDB` đã được cập nhật hoặc chèn vào bảng `dbm.dbo.staff` với đầy đủ mã nhân viên, ngày hết hạn và mức lương. Nhờ vậy, những nhân viên sẵn sàng chấm công đã có trước khi tiến hành các bước tiếp theo.

- Tiếp đó, phần ETL TimeSheet sẽ ghép (MERGE) dữ liệu từ bảng OfflineDB.dbo.TimeSheet vào bảng dbm.dbo.time\_sheet. Thủ tục lấy từng mã chấm công (TimeSheet.id) từ hệ thống nguồn và tìm tương ứng mã nhân viên đích (dbm.dbo.staff.id) dựa trên quan hệ mã nhân viên chung (code). Với mỗi nhân viên chưa có bản ghi trong bảng time\_sheet của DBM, nó sẽ chèn thêm một dòng mới chỉ bao gồm staff\_id. Nhờ đó, mọi nhân viên đã được đồng bộ từ EtlStaff đều có một bản ghi chấm công trống trong DBM, sẵn sàng nhận dữ liệu chi tiết.
- Ngay sau khi đảm bảo tất cả nhân viên đã có khung chấm công, bước ETL RecordDay sẽ chuyển từng bản ghi chấm công cụ thể từ OfflineDB.dbo.RecordDay vào dbm.dbo.record\_day. Theo đó, thủ tục tập hợp các trường ngày (day), giờ vào (checkin), giờ ra (checkout), trạng thái vào muộn/đúng giờ (in\_status) và trạng thái ra sớm/đúng giờ (out\_status), đồng thời tìm ra time\_sheet\_id tương ứng trong DBM thông qua mối liên kết giữa mã nhân viên. Nếu một bản ghi đã tồn tại nhưng có bất kỳ thay đổi nào về thời gian hoặc trạng thái chấm công, thủ tục sẽ cập nhật lại cho khớp với dữ liệu nguồn. Ngược lại, nếu là một ngày mới chưa có trong DBM, nó chèn thêm tất cả thông tin chi tiết đó vào bảng record\_day.
- Cuối cùng, sau khi hoàn tất cả hai phần TimeSheet và RecordDay, thủ tục in ra thông báo “ETL TimeSheet & RecordDay completed successfully.” để xác nhận quá trình đồng bộ đã hoàn thành. Nhờ cách tổ chức này, bạn chỉ cần một lệnh gọi duy nhất để đồng bộ đầy đủ khung nhân viên và mọi dữ liệu chấm công hàng ngày từ hệ thống OfflineDB về DBM, đảm bảo tính nhất quán, đầy đủ và dễ theo dõi.

## V. Xây dựng giao diện web

### 1. Giao diện đăng nhập, đăng ký

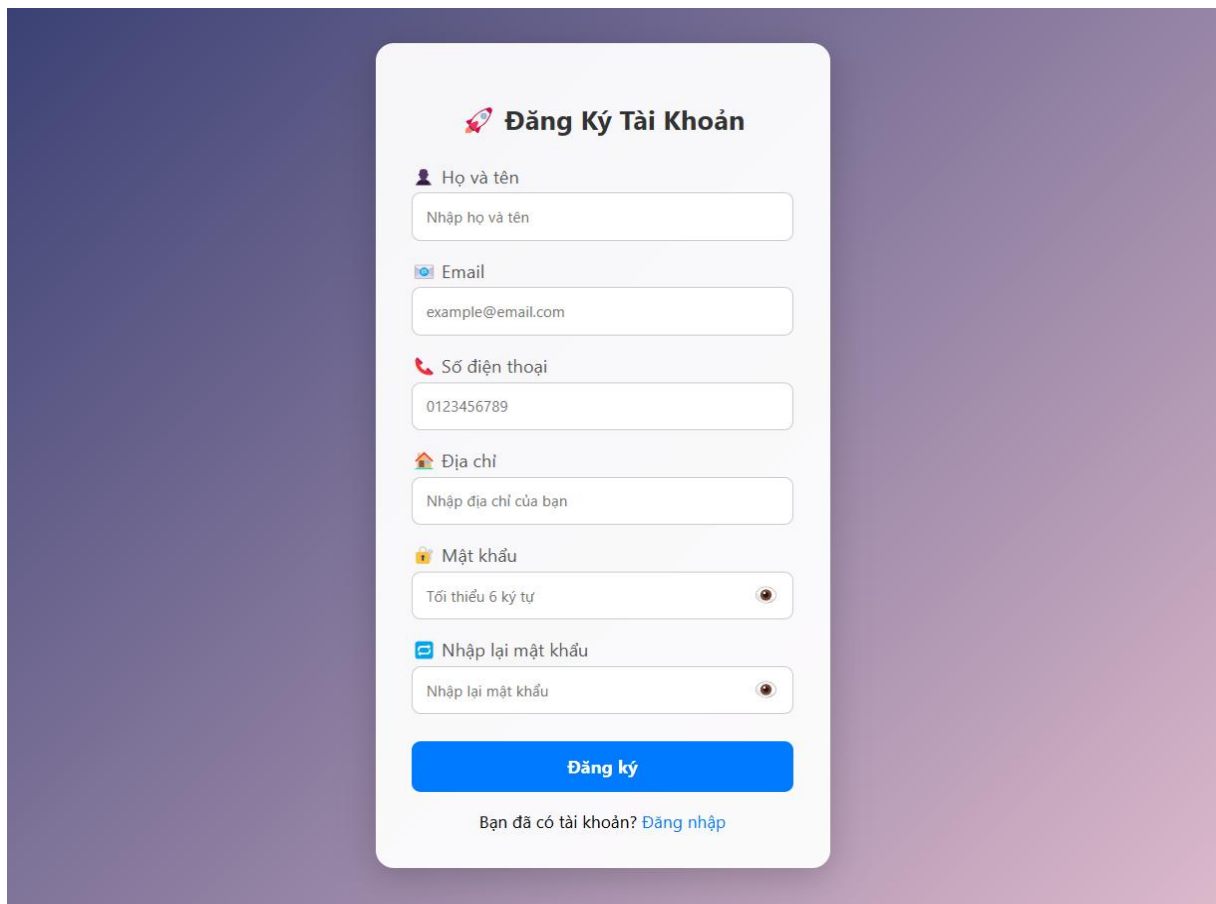
Giao diện đăng nhập: người dùng cần nhập email và mật khẩu để đăng nhập.



The image shows a login form centered on a blue gradient background. The form is a white rounded rectangle with a shadow. At the top, it has a blue hanger icon and the text "Chào mừng bạn". Below this are two input fields: the first is labeled "Email" with an envelope icon, and the second is labeled "Mật khẩu" with a lock icon. A blue button with the text "Đăng nhập" is positioned below the input fields. At the bottom of the form, there is a link that says "Bạn chưa có tài khoản? Đăng ký".



Giao diện đăng ký tài khoản cho khách hàng: khách hàng điền tên, email, số điện thoại, địa chỉ và mật khẩu để tạo tài khoản.



**Đăng Ký Tài Khoản**

Họ và tên  
Nhập họ và tên

Email  
example@email.com

Số điện thoại  
0123456789

Địa chỉ  
Nhập địa chỉ của bạn

Mật khẩu  
Tối thiểu 6 ký tự

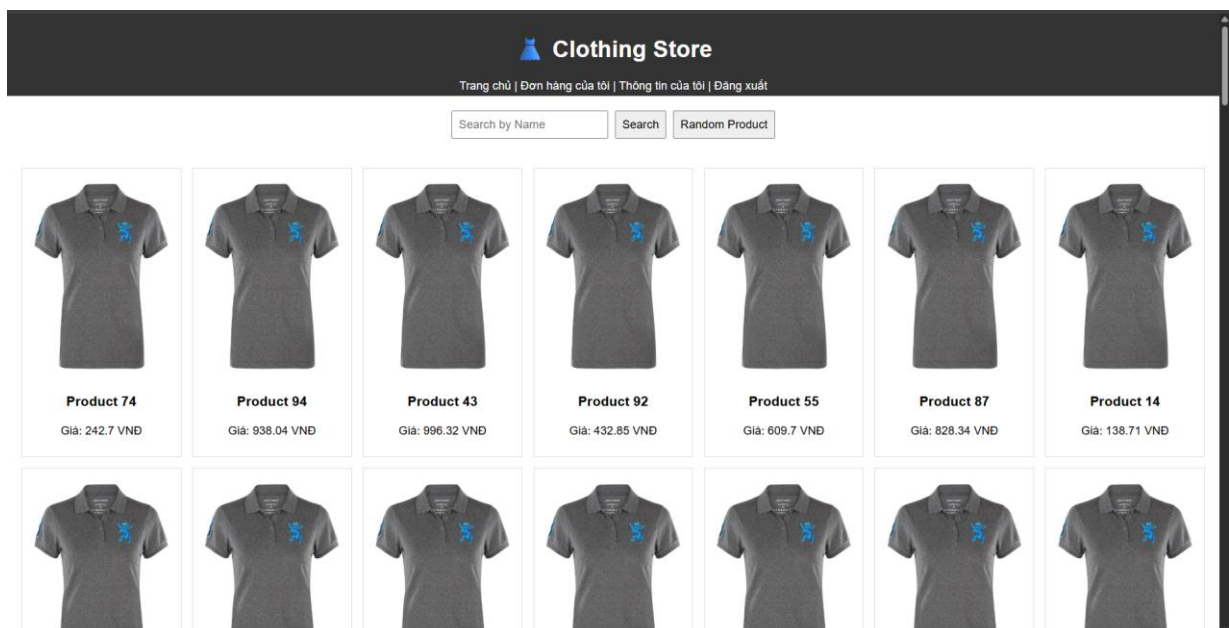
Nhập lại mật khẩu  
Nhập lại mật khẩu

**Đăng ký**

Bạn đã có tài khoản? [Đăng nhập](#)


## 2. Giao diện trang chủ mua hàng online

Giao diện trang chủ: liệt kê các sản phẩm để khách hàng lựa chọn sản phẩm cần mua. Ngoài ra còn có chức năng tìm kiếm sản phẩm theo tên sản phẩm, tìm kiếm ngẫu nhiên sản phẩm. Nhấn vào một sản phẩm để xem chi tiết sản phẩm đó và mua hàng.



### 3. Thông tin cá nhân

Trang thông tin cá nhân: cho phép người dùng xem lại thông tin cá nhân của mình, ngoài ra còn có thể cập nhật lại thông tin.

 Clothing Store

Trang chủ | Đơn hàng của tôi | Thông tin của tôi | Đăng xuất

User Information

**Name:** User 234534543

**Email:** user2@example.com

**Phone:** 0900000002

**Address:** S? 2, Đu?ng ABC, Qu?n XYZ


**Role:** CUSTOMER

**Active:** Yes


Edit Info

### 4. Trang chi tiết sản phẩm

Trang chi tiết sản phẩm: hiển thị các thông tin chi tiết của sản phẩm như: tên sản phẩm, ảnh sản phẩm, loại sản phẩm, giá của sản phẩm, nhà cung cấp, số lượng còn lại trong kho,...

 Clothing Store

Trang chủ | Đơn hàng của tôi | Thông tin của tôi | Đăng xuất



Product 55

**Danh mục:** Category 1

**Giá:** 609.7 VNĐ

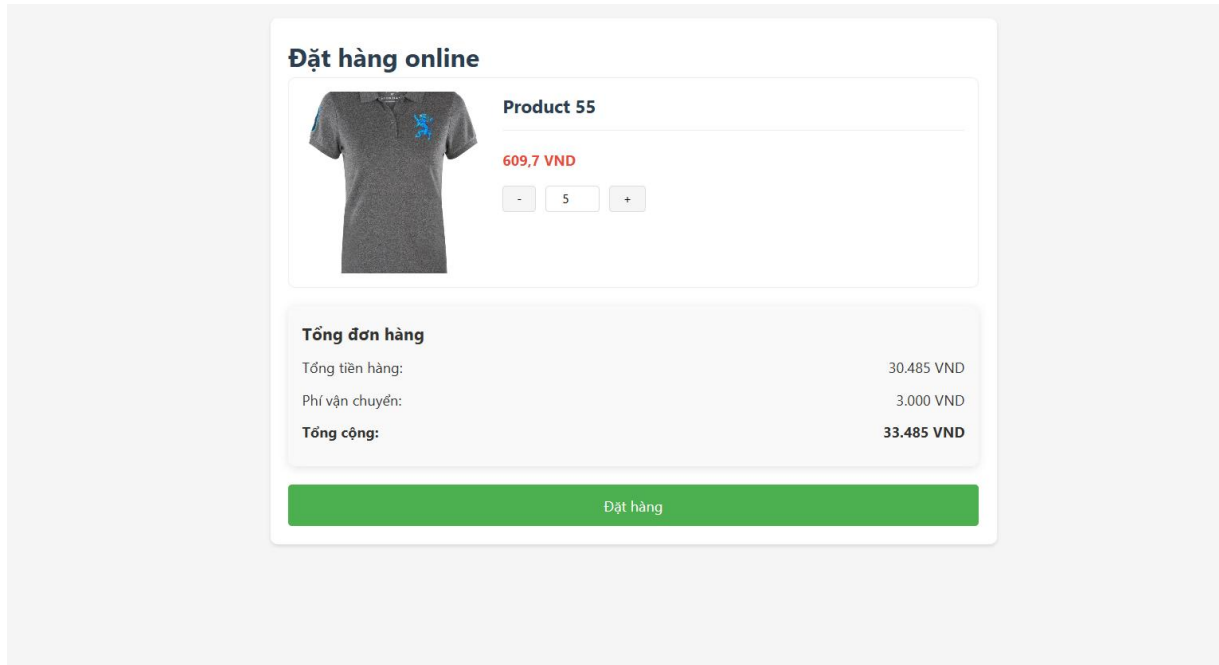
**Nhà cung cấp:** Supplier 4

**Số lượng:** 1169861

Mua

## 5. Trang thanh toán

Trang đặt hàng online: sau khi chọn một sản phẩm để mua hàng chuyển sang trang đặt hàng cho phép khách hàng chọn số lượng sản phẩm cần đặt hàng. Hiển thị các thông tin về đơn hàng sẽ tạo như: Tổng tiền hàng, phí vận chuyển,... Nhấn nút đặt hàng để đặt hàng và lưu thông tin đơn hàng vào cơ sở dữ liệu.

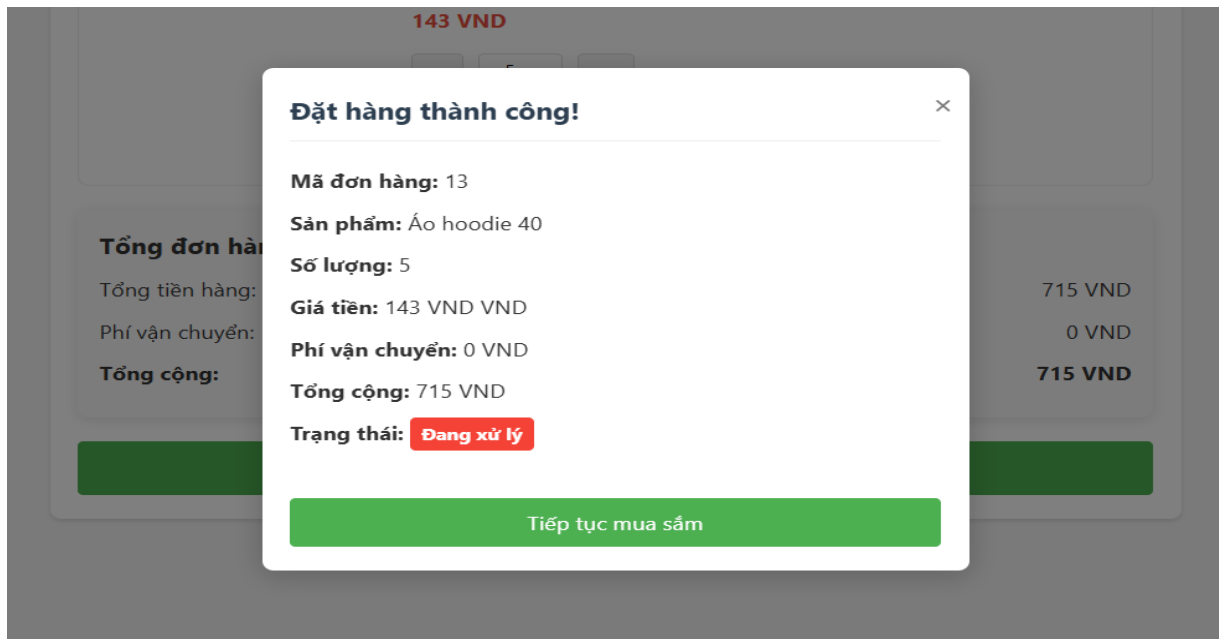


The screenshot shows a web interface for online ordering. At the top, there's a section titled "Đặt hàng online". Below it, a product image of a grey polo shirt is displayed next to the text "Product 55" and a price of "609,7 VND". There are minus, quantity (5), and plus buttons. Below the product, a summary table is shown:

Tổng đơn hàng	
Tổng tiền hàng:	30.485 VND
Phí vận chuyển:	3.000 VND
<b>Tổng cộng:</b>	<b>33.485 VND</b>

At the bottom of the summary section is a green button labeled "Đặt hàng".

Giao diện đặt hàng thành công: Hiển thị các thông tin của đơn hàng như: mã đơn hàng, các sản phẩm, số lượng, thành tiền, phí vận chuyển và trạng thái đơn hàng.




The screenshot shows a modal window titled "Đặt hàng thành công!" (Order successful!). It contains the following information:

- Mã đơn hàng: 13
- Sản phẩm: Áo hoodie 40
- Số lượng: 5
- Giá tiền: 143 VND VND
- Phí vận chuyển: 0 VND
- Tổng cộng: 715 VND
- Trạng thái: **Đang xử lý**

At the bottom of the modal is a green button labeled "Tiếp tục mua sắm".

## 6. Trang lịch sử mua hàng

Trang lịch sử mua hàng: hiển thị danh sách các đơn hàng mà khách hàng đã từng mua và đơn hàng hiện tại. Mỗi đơn hàng đều có thông tin chi tiết về đơn hàng.

<div> Clothing Store</div> <div>Trang chủ   Đơn hàng của tôi   Thông tin của tôi   Đăng xuất</div>	
Đơn hàng của tôi	
<div><div>Đơn hàng #1</div><div>Ngày đặt: 23:47 08/05/2025</div><div>Người nhận: User 2</div><div>Số điện thoại: 0900000002</div><div>Địa chỉ: S? 2, Đu?ng ABC, Qu?n XYZ</div><div>Sản phẩm:</div><div>Đ? ng? 44 x 5</div></div>	<div>Hoàn thành</div> <div>Giá trị đơn hàng: 1.852 đ</div> <div>Phí vận chuyển: 0 đ</div> <div>Tổng cộng: 1.852 đ</div>
<div><div>Đơn hàng #2</div><div>Ngày đặt: 23:47 08/05/2025</div><div>Người nhận: User 2</div><div>Số điện thoại: 0900000002</div><div>Địa chỉ: S? 2, Đu?ng ABC, Qu?n XYZ</div><div>Sản phẩm:</div><div>Váy 67 x 4</div></div>	<div>Chờ xử lý</div> <div>2.046 đ</div>

## 7. Trang tạo hóa đơn cho khách hàng

Trang tạo đơn hàng cho khách hàng: Nhân viên có thể sử dụng trang này để tìm kiếm sản phẩm theo yêu cầu của khách hàng và tiến hành tạo đơn hàng cho khách hàng đến cửa hàng trực tiếp. Sau khi tạo đơn hàng thành công sẽ hiển thị thông tin chi tiết về đơn hàng vừa tạo. Ngoài ra ở giao diện này nhân viên cũng có thể tự thực hiện chấm công cho bản thân.

Clothing Store

[Home](#) | [Login](#) | [CheckIn](#) | [CheckOut](#)

### Thanh toán tại cửa hàng

Tim kiếm sản phẩm:

Áo dài 12 - 781,92 VND

Qu?n short 20 - 910,51 VND

Qu?n tây 21 - 694,69 VND

Váy 22 - 506,75 VND

Thêm sản phẩm

Tổng tiền: 0 VND

Tạo đơn hàng

### Hóa đơn

Chi nhánh: HANOI

Mã đơn hàng: 15

Sản phẩm: Qu?n tây 6, Số lượng: 1, Giá: 251,24 VND

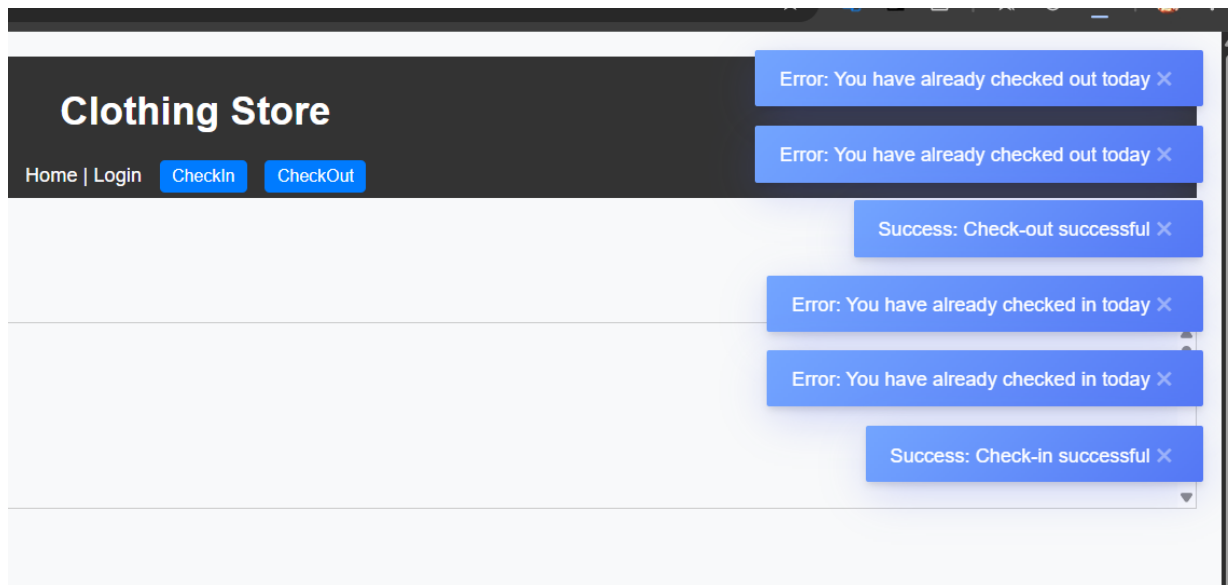
Sản phẩm: Áo so mi 16, Số lượng: 1, Giá: 632,18 VND

Tổng tiền: 883,42 VND

Ngày tạo: 00:04:25 9/5/2025

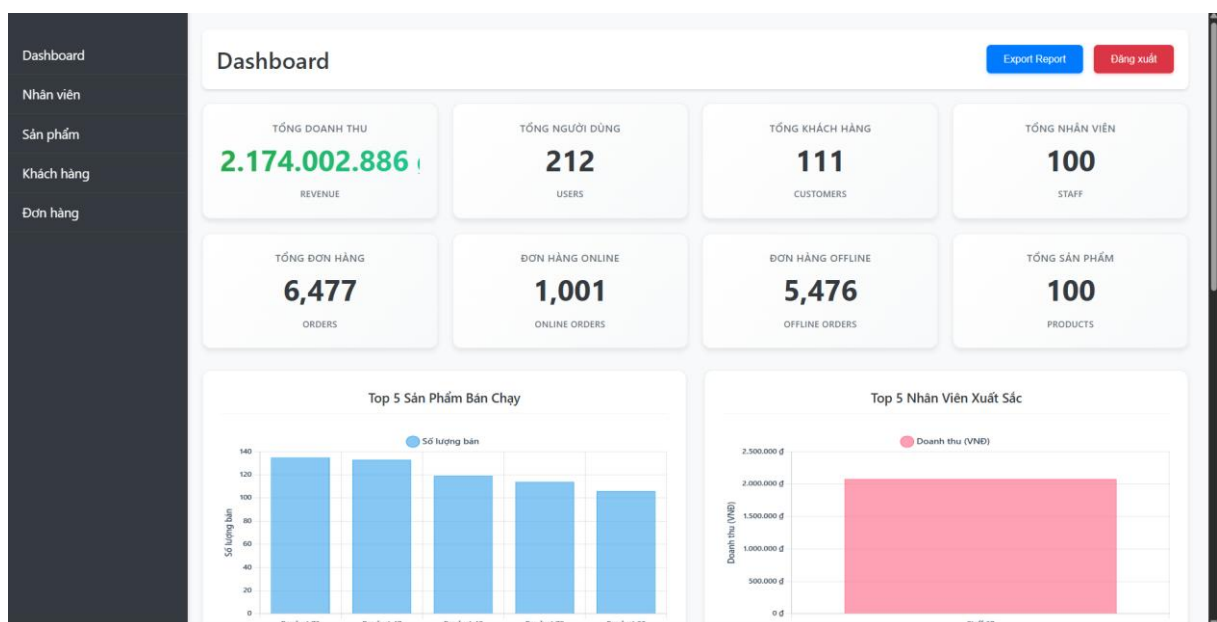
## 8. Chức năng chấm công cho nhân viên

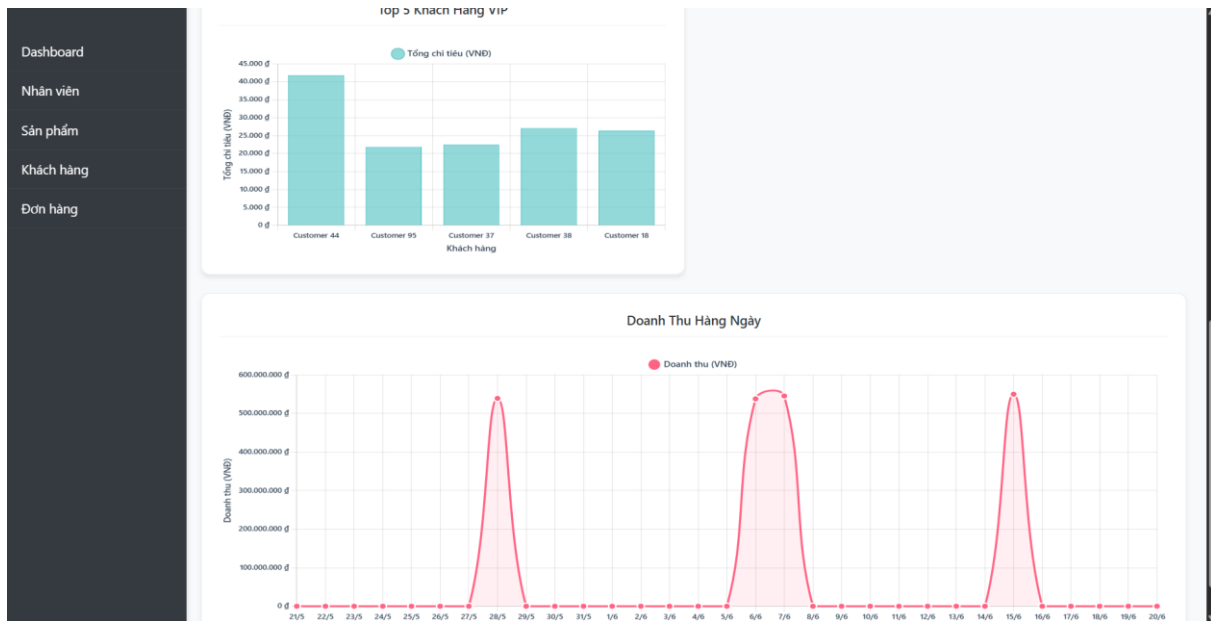
Khi nhân viên đăng nhập sẽ có các nút Checkin và Checkout trên thanh điều hướng để nhân viên thực hiện chấm công.



## 9. Trang dashboard

Trang dashboard: Admin có thể truy cập trang này để xem thông tin tổng quan của cửa hàng. Các thông tin có thể xem bao gồm: Tổng số người dùng, số khách hàng, số nhân viên, số sản phẩm hiện tại, số đơn hàng (online, offline), các biểu đồ về Doanh thu theo ngày, tháng, top khách hàng mua nhiều nhất, top nhân viên, top sản phẩm bán chạy nhất,... Ngoài ra còn có chức năng xuất báo cáo theo mẫu tạo sẵn.





## 10. Trang quản lý nhân viên

Trang quản lý nhân viên: hiển thị danh sách toàn bộ nhân viên của cửa hàng. Admin có thể thêm nhân viên mới, tìm kiếm lọc nhân viên theo tên, theo chi nhánh, theo trạng thái,...

Mỗi dòng ứng với một nhân viên với các thông tin cơ bản như id, tên, email, số điện thoại,... Có thể nhấn nút Xem để xem chi tiết nhân viên và cập nhật thông tin hoặc cập nhật trạng thái tài khoản của nhân viên (hoạt động hoặc ngừng hoạt động).

Dashboard
Nhân viên
Sản phẩm
Khách hàng
Đơn hàng

### Quản lý nhân viên

Tất cả chi nhánh
Tất cả trạng thái

Tìm kiếm
Đặt lại

Thêm nhân viên

ID	Tên	Email	Số điện thoại	Trạng thái	Chi nhánh	Mã nhân viên	Thao tác
1	User 21	user21@example.com	0900000021	Ngừng hoạt động	ONLINE	STF0001	Xem
2	User 22	user22@example.com	0900000022	Đang hoạt động	HANOI	STF0002	Xem
3	User 23	user23@example.com	0900000023	Ngừng hoạt động	ONLINE	STF0003	Xem
4	User 24	user24@example.com	0900000024	Đang hoạt động	HANOI	STF0004	Xem
5	User 25	user25@example.com	0900000025	Ngừng hoạt động	ONLINE	STF0005	Xem
6	User 26	user26@example.com	0900000026	Đang hoạt động	HANOI	STF0006	Xem
7	User 27	user27@example.com	0900000027	Ngừng hoạt động	ONLINE	STF0007	Xem
8	User 28	user28@example.com	0900000028	Đang hoạt động	HANOI	STF0008	Xem
9	User 29	user29@example.com	0900000029	Ngừng hoạt động	ONLINE	STF0009	Xem
10	User 30	user30@example.com	0900000030	Đang hoạt động	HANOI	STF0010	Xem

## Giao diện thêm nhân viên:

**Chi tiết nhân viên**

ID nhân viên: 1  
Tên: Staff 1  
Email: staff1@example.com  
Số điện thoại: 01  
Địa chỉ: Address 1  
Trạng thái: **Đang hoạt động**  
Chi nhánh: HANOI  
Mã nhân viên: Staff1  
Ngày hết hạn: 28/9/2025  
Lương: 1.237,48 VND

Sửa Hủy

ID	Tên	Email	Số điện thoại	Trạng thái	Chi nhánh	Mã nhân viên	Thao tác
1	Staff 1	staff1@example.com	01	Đang hoạt động	HANOI	Staff1	Xem
2	Staff 10	staff10@example.com	010	Đang hoạt động	HANOI	Staff10	Xem
3	Staff 100	staff100@example.com	0100	Đang hoạt động	HANOI	Staff100	Xem
4	Staff 11	staff11@example.com	011	Đang hoạt động	HANOI	Staff11	Xem
5	Staff 12	staff12@example.com	012	Đang hoạt động	HANOI	Staff12	Xem
6	Staff 13	staff13@example.com	013	Đang hoạt động	HANOI	Staff13	Xem
7	Staff 14	staff14@example.com	014	Đang hoạt động	HANOI	Staff14	Xem
8	Staff 15	staff15@example.com	015	Đang hoạt động	HANOI	Staff15	Xem
9	Staff 16	staff16@example.com	016	Đang hoạt động	HANOI	Staff16	Xem
10	Staff 17	staff17@example.com	017	Đang hoạt động	HANOI	Staff17	Xem
11	Staff 18	staff18@example.com	018	Đang hoạt động	HANOI	Staff18	Xem
12	Staff 19	staff19@example.com	019	Đang hoạt động	HANOI	Staff19	Xem

## Giao diện cập nhật thông tin nhân viên:

**Chỉnh sửa thông tin nhân viên**

Tên: Staff 1  
Email: staff1@example.com  
Số điện thoại: 01  
Địa chỉ: Address 1  
Chi nhánh: HANOI  
Ngày hết hạn: 2025-09-28  
Lương: 1237.48

Lưu Hủy

ID	Tên	Email	Số điện thoại	Trạng thái	Chi nhánh	Mã nhân viên	Thao tác
1	Staff 1	staff1@example.com	01	Đang hoạt động	HANOI	Staff1	Xem
2	Staff 10	staff10@example.com	010	Đang hoạt động	HANOI	Staff10	Xem
3	Staff 100	staff100@example.com	0100	Đang hoạt động	HANOI	Staff100	Xem
4	Staff 11	staff11@example.com	011	Đang hoạt động	HANOI	Staff11	Xem
5	Staff 12	staff12@example.com	012	Đang hoạt động	HANOI	Staff12	Xem
6	Staff 13	staff13@example.com	013	Đang hoạt động	HANOI	Staff13	Xem
7	Staff 14	staff14@example.com	014	Đang hoạt động	HANOI	Staff14	Xem
8	Staff 15	staff15@example.com	015	Đang hoạt động	HANOI	Staff15	Xem
9	Staff 16	staff16@example.com	016	Đang hoạt động	HANOI	Staff16	Xem
10	Staff 17	staff17@example.com	017	Đang hoạt động	HANOI	Staff17	Xem
11	Staff 18	staff18@example.com	018	Đang hoạt động	HANOI	Staff18	Xem
12	Staff 19	staff19@example.com	019	Đang hoạt động	HANOI	Staff19	Xem
13	Staff 2	staff2@example.com	020	Đang hoạt động	HANOI	Staff2	Xem
14	Staff 20	staff20@example.com	020	Đang hoạt động	HANOI	Staff20	Xem



11. Xuất báo cáo

Trong giao diện dashboard có thể xuất báo cáo theo mẫu có sẵn theo các dữ liệu từ trang dashboard hiện tại và người xuất báo cáo.

CHUỖI CỬA HÀNG QUẦN ÁO ABC

Địa chỉ: 123/3 đường Lê Lợi, phường Bến Nghé, Quận 1, Thành phố Hồ Chí Minh

MST: xxxxxxxxxxxx

BÁO CÁO TÀI CHÍNH TỔNG HỢP

Người xuất báo cáo: Admin

Ngày xuất: 19/06/2025

I. THỐNG KÊ DOANH THU

Đơn vị tính: VNĐ

Tiêu chí	Giá trị
1	2
Tổng doanh thu	2,174,002,886 VNĐ

II. THỐNG KÊ ĐƠN HÀNG

Đơn vị tính: Đơn hàng

Tiêu chí	Số lượng
1	2
Tổng số đơn hàng	6477
Đơn hàng online	1001
Đơn hàng offline	5476

III. THỐNG KÊ NGƯỜI DÙNG

Đơn vị tính: Người/Sản phẩm

Tiêu chí	Số lượng
1	2
Tổng người dùng	212
Tổng số khách hàng	111
Tổng số nhân viên	100
Tổng số sản phẩm	100

#### IV. TOP 5 SẢN PHẨM BÁN CHẠY NHẤT

Đơn vị tính: Cái/VNĐ

STT	Tên sản phẩm	Số lượng đã bán	Doanh thu
1	2	3	4
1	Product 73	135	48,277 VNĐ
2	Product 47	133	132,511 VNĐ
3	Product 40	119	105,264 VNĐ
4	Product 79	114	5,151 VNĐ
5	Product 29	106	98,370 VNĐ

#### V. TOP 5 NHÂN VIÊN KINH DOANH

Đơn vị tính: Cái/VNĐ

STT	Mã NV	Tên nhân viên	Số lượng đã bán	Doanh thu
1	2	3	4	5
1	Staff67	Staff 67	4228	2,076,147 VNĐ

#### VI. TOP 5 KHÁCH HÀNG MUA NHIỀU NHẤT

Đơn vị tính: Cái/VNĐ

STT	Tên khách hàng	Email	SĐT	Số lượng mua	Tổng chi tiêu
1	2	3	4	5	6
1	Customer 44	email44@gmail.com	0000000044	100	41,830 VNĐ
2	Customer 95	email95@gmail.com	0000000095	64	21,848 VNĐ
3	Customer 37	email37@gmail.com	0000000037	58	22,518 VNĐ
4	Customer 38	email38@gmail.com	0000000038	53	27,082 VNĐ
5	Customer 18	email18@gmail.com	0000000018	52	26,399 VNĐ

Người lập biểu (Ký, họ tên)	Kế toán trưởng (Ký, họ tên)	Người đại diện theo pháp luật (Ký, họ tên)
Admin		
© CH-ABC 2025 - Hệ thống quản lý doanh thu		

## 12. Trang quản lý sản phẩm

Trang quản lý sản phẩm: hiển thị danh sách các sản phẩm của cửa hàng, mỗi dòng trong bảng tương ứng với một sản phẩm, và hiển thị thông tin cơ bản của sản phẩm, có các nút hành động như Xem chi tiết và Cập nhật sản phẩm để cập nhật thông tin của sản phẩm.

Admin có thể thêm sản phẩm mới bằng nút Thêm sản phẩm.

Trên cùng có một số thông tin thống kê để quản lý các sản phẩm của cửa hàng dễ dàng hơn thông qua các số liệu trực quan.

Ngoài ra còn có các chức năng lọc, tìm kiếm sản phẩm theo tên sản phẩm, số lượng sản phẩm trong kho, giá của sản phẩm,....

- Dashboard
- Nhân viên
- Sản phẩm
- Khách hàng
- Đơn hàng

### Product Management

Total Products  
100

Inventory Value  
\$318656476381.13

Highest Price  
\$996.32

Lowest Price  
\$37.84

Avg Price  
\$503.39

Out of Stock  
0

Top Category  
Category 8

Total Stock  
608971088

Search

Stock as:

Price Range:  to

Stock Status: All

Apply Filters
Reset

Add Product

ID	Product Name	Category	Supplier	Price	Branch Stocks	Actions
1	Product 1	Category 8	Supplier 12	181.54	<ul style="list-style-type: none"> <li>ONLINE 4728904</li> <li>HANOI 99798</li> </ul>	<span style="background-color: #ffc107; padding: 5px 10px; border-radius: 5px;">Update</span> <span style="background-color: #dc3545; color: white; padding: 5px 10px; border-radius: 5px;">Delete</span>
2	Product 10	Category 10	Supplier 7	285.53	<ul style="list-style-type: none"> <li>ONLINE 4516780</li> <li>HANOI 5946275</li> </ul>	<span style="background-color: #ffc107; padding: 5px 10px; border-radius: 5px;">Update</span> <span style="background-color: #dc3545; color: white; padding: 5px 10px; border-radius: 5px;">Delete</span>

### 13. Trang quản lý khách hàng

Trang quản lý khách hàng: hiển thị danh sách các khách hàng của cửa hàng, admin có thể dễ dàng tìm thấy thông tin khách hàng để liên lạc nếu cần thiết. Có thể xem chi tiết thông tin mỗi khách hàng thông qua nút hành động Xem chi tiết.

Có các tính năng tìm kiếm, lọc khách hàng theo tên, theo trạng thái,...

Admin còn có thể chặn khách hàng nếu khách hàng vi phạm chính sách của cửa hàng.

Dashboard

Nhân viên

Sản phẩm

Khách hàng

Đơn hàng

Quản lý khách hàng

Tìm kiếm khách hàng...

Tất cả trạng thái

Tìm kiếmĐặt lại

ID	Tên	Email	Số điện thoại	Địa chỉ	Trạng thái	Thao tác
1	User 234534543	user2@example.com	0900000002	S? 2, Đu?ng ABC, Qu?n XYZ	Đang hoạt động	XemBlock
2	User 3	user3@example.com	0900000003	S? 3, Đu?ng ABC, Qu?n XYZ	Ngừng hoạt động	Xem
3	User 4	user4@example.com	0900000004	S? 4, Đu?ng ABC, Qu?n XYZ	Đang hoạt động	XemBlock
4	User 5	user5@example.com	0900000005	S? 5, Đu?ng ABC, Qu?n XYZ	Ngừng hoạt động	Xem
5	User 6	user6@example.com	0900000006	S? 6, Đu?ng ABC, Qu?n XYZ	Đang hoạt động	XemBlock
6	User 7	user7@example.com	0900000007	S? 7, Đu?ng ABC, Qu?n XYZ	Ngừng hoạt động	Xem
7	User 8	user8@example.com	0900000008	S? 8, Đu?ng ABC, Qu?n XYZ	Đang hoạt động	XemBlock
8	User 9	user9@example.com	0900000009	S? 9, Đu?ng ABC, Qu?n XYZ	Ngừng hoạt động	Xem
9	User 10	user10@example.com	0900000010	S? 10, Đu?ng ABC, Qu?n XYZ	Đang hoạt động	XemBlock
10	User 11	user11@example.com	0900000011	S? 11, Đu?ng ABC, Qu?n XYZ	Ngừng hoạt động	Xem
11	User 12	user12@example.com	0900000012	S? 12, Đu?ng ABC, Qu?n XYZ	Đang hoạt động	XemBlock

Giao diện xem chi tiết thông tin của khách hàng:

Dashboard

Nhân viên

Sản phẩm

Khách hàng

Đơn hàng

Quản lý khách hàng

Tìm kiếm khách hàng...

Tất cả trạng thái

Tìm kiếmĐặt lại

Chi tiết khách hàng

ID khách hàng: 1  
Tên: Customer 1  
Email: email1@gmail.com  
Số điện thoại: 0000000001  
Địa chỉ: Address 1  
Trạng thái: Đang hoạt động

KhóaHủy

ID	Tên	Trạng thái	Thao tác
1	Customer 1	Đang hoạt động	XemBlock
2	Customer 10	Đang hoạt động	XemBlock
3	Customer 100	Đang hoạt động	XemBlock
4	Customer 101	Đang hoạt động	XemBlock
5	Customer 102	Đang hoạt động	XemBlock
6	Customer 103	Đang hoạt động	XemBlock
7	Customer 104	Đang hoạt động	XemBlock
8	Customer 105	Đang hoạt động	XemBlock
9	Customer 106	Đang hoạt động	XemBlock
10	Customer 107	Đang hoạt động	XemBlock
11	Customer 108	Đang hoạt động	XemBlock

## 14. Trang quản lý đơn hàng

Trang quản lý đơn hàng: liệt kê danh sách các đơn hàng của cửa hàng. Admin có thể dễ dàng xem các thông tin của mỗi đơn hàng để chuẩn bị hàng giao cho khách.

Admin có thể cập nhật trạng thái của đơn hàng sang các trạng thái tiếp theo hoặc hủy đơn hàng.

Có đầy đủ các chức năng lọc, tìm kiếm, đơn hàng theo tên, loại, trạng thái, ngày đặt hàng,...

Dashboard

Nhân viên

Sản phẩm

Khách hàng

Đơn hàng

Quản lý đơn hàng

Tìm kiếm đơn hàng...

Tất cả loại

Tất cả trạng thái

Từ ngày: yyyy-mm-dd Đến ngày: yyyy-mm-dd

Tim kiếm

Đặt lại

Chi nhánh	Mã đơn hàng	Loại	Khách hàng	Số điện thoại	Nhân viên	Trạng thái	Ngày tạo	Tổng tiền	Thao tác
N/A	1	ONLINE	User 234534543	0900000002	N/A	Hoàn thành	8/5/2025 23:47:06	1.851,8 VND	Xem
N/A	2	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:13	2.045,64 VND	Xem
N/A	3	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:18	6.424,96 VND	Xem
N/A	4	ONLINE	User 234534543	0900000002	N/A	Đã hủy	8/5/2025 23:47:22	1.109,08 VND	Xem
N/A	5	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:26	4.862,83 VND	Xem
N/A	6	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:34	11.801,69 VND	Xem
N/A	7	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:39	8.391,9 VND	Xem
HANOI	8	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:51:55	8.829,45 VND	Xem
HANOI	9	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:52:30	17.386,88 VND	Xem
HANOI	10	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:52:52	3.632,16 VND	Xem
HANOI	11	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:53:31	4.974,62 VND	Xem

Giao diện xem chi tiết đơn hàng:

Dashboard

Nhân viên

Sản phẩm

Khách hàng

Đơn hàng

Quản lý đơn hàng

Tìm kiếm đơn hàng...

Tất cả loại

Tất cả trạng thái

Từ ngày: yyyy-mm-dd Đến ngày: yyyy-mm-dd

Tim kiếm

Đặt lại

Chi nhánh	Mã đơn hàng	Loại	Khách hàng	Số điện thoại	Nhân viên	Trạng thái	Ngày tạo	Tổng tiền	Thao tác
N/A	7	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:06	1.851,8 VND	Xem
N/A	2	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:13	2.045,64 VND	Xem
N/A	3	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:18	6.424,96 VND	Xem
N/A	4	ONLINE	User 234534543	0900000002	N/A	Đã hủy	8/5/2025 23:47:22	1.109,08 VND	Xem
N/A	5	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:26	4.862,83 VND	Xem
N/A	6	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:34	11.801,69 VND	Xem
N/A	7	ONLINE	User 234534543	0900000002	N/A	Đang xử lý	8/5/2025 23:47:39	8.391,9 VND	Xem
HANOI	8	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:51:55	8.829,45 VND	Xem
HANOI	9	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:52:30	17.386,88 VND	Xem
HANOI	10	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:52:52	3.632,16 VND	Xem
HANOI	11	OFFLINE	Khách lẻ	N/A	oki lên	Hoàn thành	8/5/2025 23:53:31	4.974,62 VND	Xem

Chi tiết đơn hàng

Mã đơn hàng: 4001

Loại đơn hàng: OFFLINE

Chi nhánh: HANOI

Khách hàng: Khách lẻ

Số điện thoại: N/A

Nhân viên: Staff 85

Trạng thái: Đang xử lý

Ngày tạo: 20/6/2025 17:45:20

Sản phẩm

Sản phẩm	Số lượng	Đơn giá	Thành tiền
Không có dữ liệu			

Tổng tiền hàng: 1.346,25 VND

Phi vận chuyển: 0 VND

Tổng thanh toán: 1.346,25 VND

Cập nhật trạng thái

Hủy đơn hàng

## **VI. Cài đặt**

### **1. Khởi tạo môi trường**

- Cài đặt JDK từ phiên bản 17 trở lên.
- Cài đặt SQL server phiên bản Developer 2022 (để bao gồm cả SQL Server Agent)
- Cài đặt IDE để code backend (java) và frontend (html, css, javascript) như IntelliJ, Netbean, Visual Studio Code,...

### **2. Triển khai hệ thống**

- Mở thư mục mã nguồn trong IDE đã tải để build và chạy hệ thống. Lúc này cơ sở dữ liệu trung tâm đã được khởi tạo nhưng chưa có dữ liệu gì. Có thể bắt đầu truy cập vào hệ thống trên web để đưa vào các dữ liệu đầu tiên.
- SQL Server Management Studio (SSMS) chạy các file CreateOfflineDB.sql, CreateOnlineDB.sql, CreateOutUser.sql để khởi tạo các cơ sở dữ liệu cho các chi nhánh.
- Tiếp theo chạy các file .sql trong thư mục etl/ để tạo các procedure etl
- Sử dụng SQL Server Agent để tạo các job theo lịch trình định trước cho các procedure etl đó.

## VII. Đánh giá điểm các thành viên và ký tên

Mã sinh viên – Họ và tên	Nội dung thực hiện	Điểm	Ký tên
B22DCKH004 – Ngô Việt Anh	- Demo trang quản lý khách hàng		
B22DCKH040 – Nguyễn Hải Hiếu	- Thiết kế cơ sở dữ liệu - Tạo procedure về user - Demo trang liên quan tới đơn hàng - Thiết kế trang dashboard - Viết báo cáo	10	
B22DCKH044 – Trần Bá Hoàng	- Demo trang quản lý sản phẩm	8	
B22DCKH088 – Lê Đăng Phúc	- Thiết kế cơ sở dữ liệu - Tạo procedure về order - Làm backend cho giao diện demo - Thiết kế trang dashboard - Viết báo cáo	10	
B22DCKH108 – Nguyễn Đình Tiến	- Thiết kế cơ sở dữ liệu - Tạo procedure về product - Giao diện trang chủ, sản phẩm - Giao diện quản lý nhân viên - Xuất báo cáo - Viết báo cáo	10	