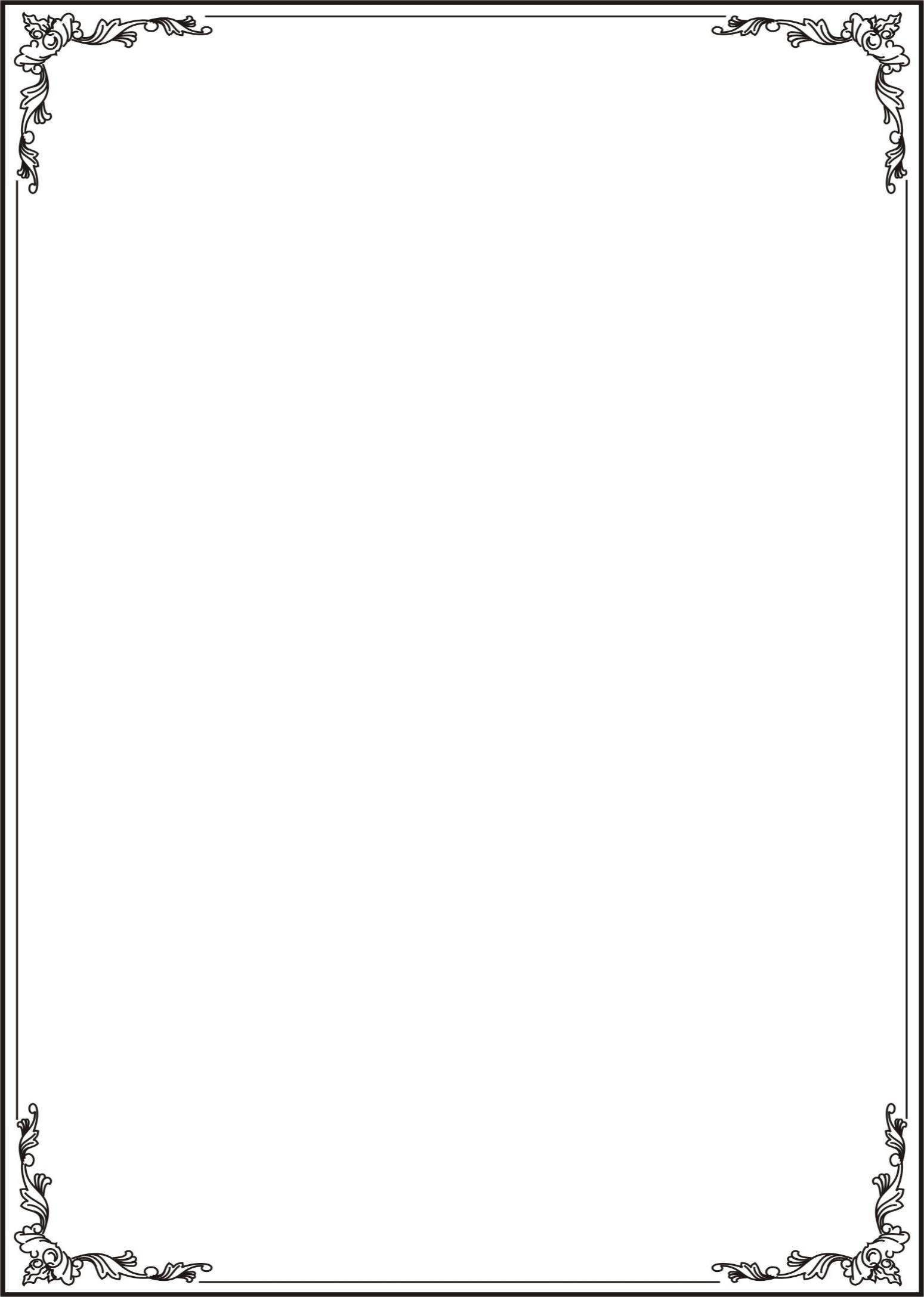
****

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**🙠🙟🕮🙝🙢**

****

**ĐỀ TÀI CUỐI KỲ MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**TÊN ĐỀ TÀI: XÂY DỰNG HỆ THỐNG QUẢN LÝ THU CHI – TÀI CHÍNH CỦA TRUNG TÂM GIỚI THIỆU VIỆC LÀM SINH VIÊN**

**GVHD:** *TS. Nguyễn Thành Sơn*

**Lớp học phần**: *251DBMS330284\_02*

**Cá nhân thực hiện:** Huỳnh Ngọc Thắng

**Nhóm:** *Nhóm 16*

**Học kỳ:** 1

**Năm học:** 2025 – 2026

Thành phố Hồ Chí Minh, tháng 10, năm 2025

**DANH SÁCH SINH VIÊN THỰC HIỆN**

**HỌC KÌ 1 NĂM HỌC 2025 – 2026**

**Nhóm 16**

**ĐỀ TÀI: XÂY DỰNG HỆ THỐNG QUẢN LÝ THU CHI – TÀI CHÍNH CỦA TRUNG TÂM GIỚI THIỆU VIỆC LÀM SINH VIÊN**

|  |  |  |
| --- | --- | --- |
| **MSSV** | **Họ và tên** | **Nhiệm vụ** |
| 23110327 | Huỳnh Ngọc Thắng | Quản lý thu chi – tài chính |

**Nhận xét:** …………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**NHẬN XÉT CỦA GIÁO VIÊN**

Tp. Hồ Chí Minh, ngày 6 tháng 10 năm 2025

**Giảng viên chấm điểm**

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN VỀ HỆ THỐNG 1](#_Toc210485705)

[1. Đặc tả đề bài 1](#_Toc210485706)

[1.1. Mô tả bài toán 1](#_Toc210485707)

[1.2. Mô tả dữ liệu 2](#_Toc210485708)

[1.3. Mô tả chức năng 4](#_Toc210485709)

[1.4. Về phần giao diện 6](#_Toc210485710)

[CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG 9](#_Toc210485711)

[1. Thiết kế cơ sở dữ liệu mức quan niệm 9](#_Toc210485712)

[2. Thiết kế cơ sở dữ liệu mức logic 11](#_Toc210485713)

[3. Các ràng buộc cần có 12](#_Toc210485714)

[4. Cài đặt SQL và các ràng buộc 15](#_Toc210485715)

[5. Lược đồ quan hệ sau khi cài đặt SQL 19](#_Toc210485716)

[19](#_Toc210485717)

[6. Các view 20](#_Toc210485718)

[7. Các trigger 22](#_Toc210485719)

[CHƯƠNG 3: THIẾT KẾ CÁC CHỨC NĂNG 26](#_Toc210485720)

[1. Kết nối cơ sở dữ liệu 26](#_Toc210485721)

[2. Chức năng riêng của nhân viên tài chính 28](#_Toc210485722)

[3. Chức năng riêng của trưởng phòng tài chính 33](#_Toc210485723)

[4. Chức năng chung của trưởng phòng tài chính và kế toán 46](#_Toc210485724)

[5. Chức năng chung của nhân viên tài chính, trưởng phòng và kế toán 51](#_Toc210485725)

[CHƯƠNG 4: PHÂN QUYỀN 53](#_Toc210485726)

[1. Tạo role 53](#_Toc210485727)

[2. Phân quyền 53](#_Toc210485728)

[CHƯƠNG 5: TRANSACTION 58](#_Toc210485729)

[CHƯƠNG 6: THIẾT KẾ GIAO DIỆN 68](#_Toc210485730)

[1. Môi trường cài đặt và công nghệ sử dụng 68](#_Toc210485731)

[2. Phần giao diện đăng nhập 68](#_Toc210485732)

[3. Phần giao diện của nhân viên tài chính 68](#_Toc210485733)

[4. Phần giao diện của trưởng phòng tài chính 70](#_Toc210485734)

[5. Phần giao diện của kế toán 74](#_Toc210485735)

**LỜI CẢM ƠN**

Em xin bày tỏ lòng biết ơn chân thành đến thầy Nguyễn Thành Sơn – người đã trực tiếp giảng dạy, hướng dẫn và luôn tận tâm hỗ trợ em trong suốt quá trình học tập và thực hiện đồ án *“Xây dựng hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên”* của môn Hệ quản trị cơ sở dữ liệu.

Nhờ có sự chỉ dẫn tận tình và sự khuyến khích, tạo điều kiện của Thầy, em không chỉ tiếp thu được những kiến thức quý giá về chuyên môn mà còn học được cách tư duy logic, tiếp cận vấn đề một cách khoa học và thực tế hơn.

Đồ án lần này không chỉ là cơ hội để em áp dụng những lý thuyết của môn học mà còn là dịp để em hiểu sâu sắc hơn về cách thiết kế, sáng tạo và cách vận hành một hệ thống cơ sở dữ liệu.

Em xin trân trọng cảm ơn Thầy vì đã đồng hành, định hướng và luôn sẵn lòng chia sẻ kiến thức, kinh nghiệm để bản thân em có thể hoàn thành đề tài một cách hiệu quả nhất. Kính chúc Thầy dồi dào sức khỏe, luôn tràn đầy nhiệt huyết và gặt hái nhiều thành công hơn nữa trong sự nghiệp trồng người cao quý.

Trân trọng,

Huỳnh Ngọc Thắng.

**LỜI NÓI ĐẦU**

Kính gửi quý thầy cô và các bạn,

Đồ án **“Xây dựng hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên”**là kết quả của quá trình học tập, thực hành và ứng dụng kiến thức môn Hệ quản trị cơ sở dữ liệu cùng các học phần lập trình ứng dụng. Đây không chỉ là một bài tập lớn mà còn là cơ hội để bản thân thử thách việc xây dựng một giải pháp thực tiễn, giải quyết bài toán quản lý tài chính - một lĩnh vực đòi hỏi sự chính xác, minh bạch và bảo mật cao.

Trong suốt quá trình triển khai, bản thân em đã đối mặt với nhiều thách thức từ việc thiết kế mô hình dữ liệu, xây dựng hệ thống cho tới việc phát triển phần mềm. Tuy nhiên, với sự hướng dẫn tận tình của quý thầy cô và nỗ lực không ngừng nghỉ của bản thân, đồ án đã được hoàn thiện với đầy đủ các chức năng cốt lõi.

Trong bối cảnh chuyển đổi số đang diễn ra mạnh mẽ tại Việt Nam, nhu cầu số hóa quy trình quản lý tài chính của doanh nghiệp ngày càng trở nên cấp thiết. Thay vì ghi chép thủ công và sử dụng bảng tính Excel đơn giản, các doanh nghiệp cần một hệ thống tập trung, có khả năng theo dõi dòng tiền theo thời gian thực, phân bổ chi phí theo từng dự án, kiểm soát quy trình phê duyệt và tạo báo cáo phân tích nhanh chóng. Đề tài **“Xây dựng hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên”**ra đời nhằm đáp ứng nhu cầu đó, giúp trung tâm minh bạch hóa dòng tiền, giảm thiểu sai sót thủ công, tăng cường kiểm soát nội bộ và hỗ trợ ra quyết định kịp thời dựa trên số liệu chính xác.

Đồ án không chỉ giúp em củng cố kiến thức về hệ quản trị cơ sở dữ liệu mà còn rèn luyện các kỹ năng thiết yếu. Đây là hành trang quý giá giúp em tự tin hơn khi bước vào thực tế nghề nghiệp.

Em xin chân thành cảm ơn quý thầy cô đã định hướng và hỗ trợ nhiệt tình trong suốt quá trình thực hiện đồ án. Em hy vọng đồ án này sẽ là nguồn tham khảo hữu ích và truyền cảm hứng cho các bạn sinh viên trong việc ứng dụng kiến thức hệ quản trị cơ sở dữ liệu vào giải quyết các bài toán thực tiễn.

# **CHƯƠNG 1: TỔNG QUAN VỀ HỆ THỐNG**

## **1. Đặc tả đề bài**

### **1.1. Mô tả bài toán**

Hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên giải quyết bài toán quản lý dòng tiền một cách minh bạch, chính xác và hiệu quả thông qua việc liên kết giữa các vai trò trong quy trình tài chính. Từ đó, hệ thống tạo nên các nhóm đối tượng chính như sau:

Nhân viên tài chính: đây là nhóm đối tượng trực tiếp thực hiện các nghiệp vụ tài chính hằng ngày của trung tâm. Nhân viên tài chính chịu trách nhiệm tạo và quản lý các giao dịch thu – chi phát sinh từ các hoạt động và mỗi giao dịch được ghi nhận với đầy đủ thông tin trong hệ thống.

Trưởng phòng tài chính: đây là đối tượng có vai trò quản lý và giám sát toàn bộ hoạt động tài chính. Trưởng phòng chịu trách nhiệm xem xét các giao dịch do nhân viên tài chính tạo ra, đảm bảo tính hợp lệ và phù hợp với quy định trước khi phê duyệt. Ngoài ra, trưởng phòng còn quản lý toàn bộ danh mục hệ thống và là cầu nối giữa các bộ phận, đảm bảo quy trình tài chính diễn ra minh bạch, tuân thủ và hiệu quả. Đặc biệt, trưởng phòng còn có quyền truy cập các báo cáo tổng hợp theo tháng và báo cáo chi tiết theo thời gian, dự án để hỗ trợ việc ra quyết định chiến lược.

Kế toán: nhóm đối tượng này đảm nhận vai trò theo dõi, đối chiếu và kiểm tra tính chính xác của các số liệu tài chính. Kế toán có thể xem toàn bộ lịch sử giao dịch, theo dõi số dư các tài khoản ngân hàng và truy cập các báo cáo chi tiết phục vụ công tác quyết toán, tập trung vào việc giám sát, đối chiếu và đảm bảo tính nhất quán của dữ liệu tài chính. Đây là vai trò quan trọng trong việc phát hiện sai sót, hỗ trợ kiểm toán nội bộ và cung cấp thông tin chính xác cho ban lãnh đạo trung tâm.

Mục tiêu của hệ thống là số hóa toàn bộ quy trình quản lý thu – chi tài chính của trung tâm một cách tự động, chuẩn hóa và minh bạch. Hệ thống không chỉ giúp ghi nhận giao dịch theo từng tài khoản ngân hàng và dự án cụ thể, mà còn thiết lập quy trình phê duyệt rõ ràng nhằm tăng cường kiểm soát nội bộ. Hơn hết, thông qua các báo cáo thống kê theo tháng, báo cáo chi tiết theo thời gian và dự án cùng biểu đồ trực quan, trung tâm có thể dễ dàng theo dõi và đánh giá hiệu quả từng dự án, lập kế hoạch ngân sách và đưa ra quyết định điều hành kịp thời. Qua đó, hệ thống giúp giảm thiểu sai sót thủ công, tăng tốc độ xử lý nghiệp vụ, đồng thời mang lại sự minh bạch, truy vết cao và hỗ trợ công tác quản trị tài chính chuyên nghiệp cho Trung tâm giới thiệu việc làm sinh viên.

### **1.2. Mô tả dữ liệu**

**1.2.1. Mô tả tập thực thể**

Hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên đáp ứng các yêu cầu cơ bản về lưu trữ, xử lý và quản lý dữ liệu tài chính. Trong hệ thống này, giao dịch đóng vai trò trung tâm, liên kết trực tiếp với nhiều nghiệp vụ khác như quản lý tài khoản ngân hàng, phân bổ theo dự án, phân quyền theo vai trò và quy trình phê duyệt.

Mỗi nhân viên sẽ được lưu trữ thông tin cá nhân bao gồm: mã nhân viên, họ tên, email, số điện thoại và liên kết với tài khoản đăng nhập. Đây là thông tin quan trọng giúp quản lý hồ sơ nhân sự và là cơ sở dữ liệu để kết nối với các nghiệp vụ tài chính khác. Để đảm bảo tính an toàn và bảo mật thông tin, mỗi nhân viên được cấp một tài khoản đăng nhập vào hệ thống. Tài khoản này bao gồm các thông tin: tên đăng nhập, mật khẩu và trạng thái tài khoản (hoạt động hay đã khóa). Nhờ đó, chỉ những nhân viên hợp lệ mới có thể sử dụng hệ thống để thực hiện công việc.

Từ nhân viên hình thành ba vai trò chuyên biệt: Nhân viên tài chính (tạo và quản lý giao dịch), Kế toán (đối chiếu, kiểm tra và xem báo cáo chi tiết), và Trưởng phòng tài chính (phê duyệt giao dịch, quản lý danh mục, truy cập báo cáo tổng hợp). Mỗi vai trò này có các thuộc tính và quyền hạn riêng biệt, giúp hệ thống kiểm soát chặt chẽ phân quyền và đảm bảo quy trình nghiệp vụ diễn ra đúng trình tự.

Hệ thống duy trì danh mục Loại giao dịch để phân loại và chuẩn hóa các nghiệp vụ tài chính. Mỗi loại giao dịch được mô tả bởi mã loại, tên loại, nhóm (THU/CHI) và mô tả chi tiết. Danh mục này được sử dụng khi ghi nhận giao dịch và là cơ sở để tạo báo cáo thống kê theo từng nhóm thu – chi một cách chính xác.

Song song với loại giao dịch, hệ thống quản lý Tài khoản ngân hàng với các thông tin: mã tài khoản, tên tài khoản, số tài khoản, ngân hàng, số dư hiện tại và trạng thái (hoạt động/ngừng hoạt động). Số dư không được cập nhật thủ công mà chỉ thay đổi khi giao dịch được phê duyệt: tăng khi giao dịch thu được duyệt, giảm khi giao dịch chi được duyệt. Điều này đảm bảo tính toàn vẹn và minh bạch trong quản lý tài chính.

Để phục vụ theo dõi hiệu quả hoạt động, giao dịch có thể có hoặc không gắn với một Dự án. Mỗi dự án lưu trữ thông tin: mã dự án, tên dự án, ngày bắt đầu, ngày kết thúc, ngân sách và trạng thái (đang hoạt động, hoàn thành, đã vô hiệu hóa). Việc phân bổ giao dịch theo dự án giúp trung tâm dễ dàng theo dõi chi phí, đánh giá hiệu quả từng chương trình và lập kế hoạch ngân sách hợp lý.

Trung tâm của hệ thống là thực thể Giao dịch, phản ánh mọi nghiệp vụ thu hoặc chi phát sinh. Mỗi giao dịch được ghi nhận với đầy đủ thông tin: mã giao dịch, loại (THU/CHI), số tiền, ngày giao dịch, mô tả chi tiết, loại giao dịch, tài khoản ngân hàng, dự án (nếu có) và trạng thái xử lý (chờ duyệt/đã duyệt/từ chối). Đặc biệt, mỗi giao dịch lưu lại thông tin người tạo (nhân viên tài chính), người duyệt (trưởng phòng) và ngày duyệt, đảm bảo tính truy vết cao. Trước khi lưu giao dịch chi, hệ thống bắt buộc kiểm tra số dư tài khoản ngân hàng; nếu không đủ số dư, giao dịch sẽ bị từ chối ngay từ khâu tạo. Khi giao dịch được phê duyệt, hệ thống tự động hạch toán vào số dư tài khoản tương ứng, bảo đảm quy trình tài chính diễn ra chính xác và minh bạch.

Cuối cùng, từ các dữ liệu trong giao dịch và các danh mục liên quan, hệ thống sẽ hình thành nên dữ liệu phục vụ thống kê và báo cáo. Thống kê theo tháng tổng hợp các chỉ tiêu: tổng thu, tổng chi và lãi/lỗ theo từng kỳ. Báo cáo chi tiết cho phép lọc danh sách giao dịch theo khoảng thời gian và dự án cụ thể, kèm theo biểu đồ trực quan về cơ cấu thu – chi.

Tổng kết lại, hệ thống bao gồm các thực thể chính: Nhân viên (Nhân viên tài chính, Kế toán, Trưởng phòng), Tài khoản, Loại giao dịch, Tài khoản ngân hàng, Dự án, Giao dịch. Các thực thể này liên kết chặt chẽ với nhau xoay quanh quy trình "tạo – phê duyệt – hạch toán – báo cáo", giúp Trung tâm quản lý tài chính một cách minh bạch, giảm thiểu sai sót thủ công và nâng cao hiệu quả ra quyết định.

**1.2.2. Mô tả mối quan hệ**

Mỗi nhân viên sẽ sở hữu một tài khoản đăng nhập duy nhất vào hệ thống. Mối quan hệ này là 1–1, nghĩa là một nhân viên chỉ có một tài khoản và mỗi tài khoản cũng chỉ thuộc về một nhân viên.

Nhân viên chia làm 3 loại: Trưởng phòng tài chính, Kế toán và Nhân viên tài chính. Mỗi nhân viên tại một thời điểm chỉ đảm nhận đúng một vai trò nghiệp vụ (Trưởng phòng tài chính, Kế toán hoặc Nhân viên tài chính), vì mỗi nhân viên chỉ có một vai trò chuyên biệt và tại một thời điểm mỗi nhân viên thuộc đúng một vai trò, và mỗi vai trò có thể có nhiều nhân viên.

Một nhân viên tài chính có thể tạo nhiều giao dịch, nhưng mỗi giao dịch chỉ do một nhân viên tài chính tạo ra. Đây là mối quan hệ 1–N, giúp hệ thống truy vết được người chịu trách nhiệm ghi nhận từng giao dịch.

Một trưởng phòng tài chính có thể phê duyệt nhiều giao dịch, tuy nhiên mỗi giao dịch chỉ được duyệt bởi một trưởng phòng duy nhất. Đây là mối quan hệ 1–N, trong đó giao dịch chỉ có người duyệt khi đã được phê duyệt hoặc từ chối.

Một loại giao dịch có thể được sử dụng cho nhiều giao dịch khác nhau, nhưng mỗi giao dịch chỉ thuộc về một loại giao dịch duy nhất (phân nhóm THU/CHI). Đây là mối quan hệ 1–N, giúp chuẩn hóa nghiệp vụ và phục vụ báo cáo thống kê chính xác.

Một tài khoản ngân hàng có thể phát sinh nhiều giao dịch theo thời gian, nhưng mỗi giao dịch chỉ được hạch toán vào một tài khoản ngân hàng cụ thể. Đây là mối quan hệ 1–N, đảm bảo việc theo dõi dòng tiền và số dư từng tài khoản một cách rõ ràng.

Một dự án có thể phát sinh hoặc gắn với nhiều giao dịch thu hoặc chi, tuy nhiên mỗi giao dịch chỉ thuộc về một dự án hoặc không thuộc dự án nào (tùy chọn). Đây là mối quan hệ 1–N, giúp Trung tâm phân bổ chi phí, theo dõi nguồn thu và đánh giá hiệu quả từng chương trình cụ thể.

Nhờ các mối quan hệ này, hệ thống đảm bảo tính nhất quán, truy vết cao và hỗ trợ quy trình nghiệp vụ tài chính diễn ra minh bạch, chính xác từ khâu tạo giao dịch đến phê duyệt, hạch toán và báo cáo.

### **1.3. Mô tả chức năng**

Hệ thống có ba nhóm người dùng chính: Trưởng phòng tài chính, Kế toán, và Nhân viên tài chính. Ngoài ra có chức năng dùng chung cho mọi vai trò.

**Chức năng chung**

*- Đăng nhập*: Người dùng sử dụng tên đăng nhập và mật khẩu để truy cập hệ thống; hỗ trợ ẩn/hiện mật khẩu khi nhập.

**Chức năng của Nhân viên tài chính**

*- Quản lý giao dịch của tôi*: Tạo mới và chỉnh sửa giao dịch THU/CHI do chính nhân viên lập; chỉ được chỉnh sửa giao dịch ở trạng thái "chờ duyệt"; kiểm tra số dư trước khi chi; theo dõi trạng thái *chờ duyệt/đã duyệt/từ chối*.

*- Tra cứu tài khoản ngân hàng*: Xem danh sách tài khoản đang hoạt động, số tài khoản, ngân hàng và số dư hiện tại.

*- Tra cứu dự án*: Xem các dự án đang hoạt động (tên dự án, ngày bắt đầu/kết thúc, trạng thái, ngân sách).

*- Tra cứu loại giao dịch*: Xem danh mục loại giao dịch (THU/CHI) và mô tả để chọn đúng khi lập giao dịch.

**Chức năng của Kế toán**

*- Xem lịch sử giao dịch*: Xem toàn bộ giao dịch theo bộ lọc thời gian, loại, trạng thái, tài khoản ngân hàng, dự án; phục vụ đối chiếu số liệu.

*- Xem thống kê theo tháng*: Xem bảng tổng hợp *Tổng thu – Tổng chi – Lãi/Lỗ* theo tháng; biểu đồ cột/đường trực quan.

*- Xem báo cáo chi tiết theo dự án*: Lọc và xem danh sách giao dịch theo khoảng thời gian và/hoặc theo từng dự án; kèm biểu đồ cơ cấu Thu–Chi.

*- Tra cứu tài khoản ngân hàng*: Xem thông tin và số dư các tài khoản đang hoạt động.

*- Tra cứu dự án*: Xem danh mục dự án cùng trạng thái.

*- Tra cứu loại giao dịch*: Xem danh mục loại giao dịch để tham chiếu khi kiểm tra chứng từ.

**Chức năng của Trưởng phòng tài chính**

*- Phê duyệt giao dịch*: Xem, phê duyệt hoặc từ chối các giao dịch do nhân viên lập; khi duyệt, hệ thống tự động cập nhật số dư tài khoản.

*- Quản lý tài khoản nhân viên*: Tạo mới tài khoản nhân viên; khóa/mở khóa; khôi phục/kích hoạt lại tài khoản khi cần.

*- Quản lý tài khoản ngân hàng*: Thêm mới, chỉnh sửa thông tin, vô hiệu hóa tài khoản ngân hàng; số dư hiển thị chỉ để theo dõi.

*- Quản lý dự án*: Thêm, sửa thông tin dự án; thay đổi trạng thái *đang hoạt động/hoàn thành/đã vô hiệu hóa*.

*- Xem thống kê theo tháng*: Theo dõi *Tổng thu – Tổng chi – Lãi/Lỗ* theo tháng với biểu đồ tổng hợp.

*- Xem báo cáo chi tiết theo dự án*: Lọc giao dịch theo khoảng thời gian và dự án; hỗ trợ đánh giá hiệu quả từng dự án.

*- Xem lịch sử giao dịch*: Tra cứu tất cả giao dịch với đầy đủ bộ lọc (thời gian, loại, trạng thái, tài khoản, dự án).

*- Tra cứu loại giao dịch*: Xem danh mục loại giao dịch để đảm bảo thống nhất khi lập và duyệt chứng từ.

### **1.4. Về phần giao diện**

**1.4.1. Phần giao diện chung**

Hệ thống cung cấp *giao diện đăng nhập (Login\_Form)* làm cổng truy cập chính cho tất cả người dùng. Giao diện được thiết kế gọn gàng với bố cục dọc, căn giữa màn hình, bao gồm: nhãn và ô nhập Tên đăng nhập, nhãn và ô nhập Mật khẩu (mặc định ẩn ký tự), nút Ẩn/Hiện mật khẩu để kiểm soát hiển thị, nút Đăng nhập đặt ngay dưới vùng nhập liệu để thuận tiện thao tác, và nút Thoát để đóng ứng dụng. Thiết kế tập trung vào sự đơn giản và trực quan, giúp người dùng truy cập nhanh chóng vào hệ thống.

**1.4.2. Phần giao diện riêng theo vai trò**

Sau khi đăng nhập thành công, người dùng được chuyển đến trang chủ tương ứng với vai trò, nơi tập trung các chức năng công việc hằng ngày.

*Giao diện Nhân viên tài chính (TrangChuNhanVienTC\_Form)* có một cột chức năng bên trái hiển thị nhóm "Công việc" với các menu chính:

- *Quản lý giao dịch của tôi*: Mở màn hình lập và theo dõi giao dịch thu/chi cá nhân. Giao diện chia thành ba vùng: vùng lọc ở trên, bảng danh sách giao dịch ở giữa hiển thị lịch sử, và khu chi tiết ở dưới để nhập/chỉnh sửa thông tin (loại giao dịch, mã loại, số tiền, tài khoản ngân hàng, dự án, mô tả). Các nút Thêm, Sửa, Lưu, Hủy hỗ trợ thao tác linh hoạt.

**-** *Tra cứu tài khoản ngân hàng*: Hiển thị bảng danh sách các tài khoản đang hoạt động với đầy đủ thông tin: tên tài khoản, số tài khoản, ngân hàng, trạng thái và số dư hiện tại (chế độ chỉ xem).

**-** *Tra cứu dự án*: Liệt kê các dự án của trung tâm kèm thông tin chi tiết: tên dự án, ngày bắt đầu/kết thúc, ngân sách và trạng thái hoạt động (chế độ chỉ xem).

- *Tra cứu loại giao dịch*: Hiển thị danh mục chuẩn các loại thu/chi kèm mô tả, giúp nhân viên chọn đúng loại khi lập giao dịch (chế độ chỉ xem).

*Giao diện Kế toán (TrangChuKeToan\_Form)* được tổ chức theo các chức năng:

***-*** *Xem lịch sử giao dịch*: Tra cứu toàn bộ giao dịch với bộ lọc đa dạng (thời gian, loại, trạng thái, tài khoản ngân hàng, dự án, người tạo/người duyệt). Dữ liệu hiển thị dạng bảng, hỗ trợ xem chi tiết từng giao dịch để phục vụ đối chiếu số liệu.

*- Xem thống kê theo tháng*: Bảng tổng hợp các chỉ số tài chính (tổng thu, tổng chi, lãi/lỗ) theo năm/tháng, kèm biểu đồ trực quan thể hiện xu hướng.

*- Xem báo cáo chi tiết theo dự án*: Lọc giao dịch theo khoảng thời gian và/hoặc dự án cụ thể, hiển thị danh sách chi tiết kèm biểu đồ cơ cấu thu–chi để phân tích sâu.

***-*** *Tra cứu*: Các màn hình Tra cứu tài khoản ngân hàng, Tra cứu dự án, Tra cứu loại giao dịch ở chế độ chỉ xem, cung cấp thông tin tham chiếu nhanh phục vụ công tác kiểm tra và đối chiếu.

*Giao diện Trưởng phòng tài chính (TrangChuTruongPhongTC\_Form)* gồm các chức năng được chia rõ ràng theo từng nghiệp vụ:

*- Phê duyệt giao dịch*: Hiển thị danh sách giao dịch đang chờ xử lý. Khi chọn một dòng, khu chi tiết bên dưới hiển thị đầy đủ nội dung giao dịch, kèm nút Duyệt và Từ chối để ra quyết định. Khi duyệt, hệ thống tự động cập nhật số dư tài khoản tương ứng.

*- Quản lý tài khoản nhân viên*: Bảng danh sách nhân viên với các nút thao tác: Tạo mới tài khoản, Vô hiệu hóa tài khoản không còn sử dụng, Kích hoạt lại tài khoản đã khóa. Hỗ trợ tìm kiếm nhanh theo từ khóa.

*- Quản lý tài khoản ngân hàng*: Giao diện cho phép thêm mới, chỉnh sửa thông tin tài khoản, vô hiệu hóa tài khoản khi ngừng sử dụng. Số dư hiển thị chỉ để theo dõi, không cho phép chỉnh sửa thủ công.

*- Quản lý dự án*: Thêm mới, cập nhật thông tin dự án, thay đổi trạng thái. Bảng danh sách dự án hỗ trợ lọc và tìm kiếm.

*- Xem thống kê theo tháng*: Tương tự giao diện kế toán, theo dõi tổng thu – tổng chi – lãi/lỗ theo kỳ với biểu đồ tổng hợp.

*- Xem báo cáo chi tiết theo dự án*: Lọc giao dịch theo khoảng thời gian và dự án, hỗ trợ đánh giá hiệu quả tài chính từng chương trình.

*- Xem lịch sử giao dịch*: Tra cứu tất cả giao dịch với đầy đủ bộ lọc (thời gian, loại, trạng thái, tài khoản, dự án) để giám sát toàn diện.

*- Tra cứu loại giao dịch*: Xem danh mục loại giao dịch để đảm bảo thống nhất khi lập và duyệt chứng từ.

Việc xây dựng một hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên là một bước tiến quan trọng trong việc tự động hóa quy trình quản lý. Trong quá trình thiết kế, hệ thống đã sử dụng bố cục rõ ràng với sự bố trí một cách hợp lý. Các giao diện đều tối ưu cho thao tác nhanh, tránh phức tạp, đảm bảo người dùng dễ dàng tiếp cận và thực hiện công việc hiệu quả.

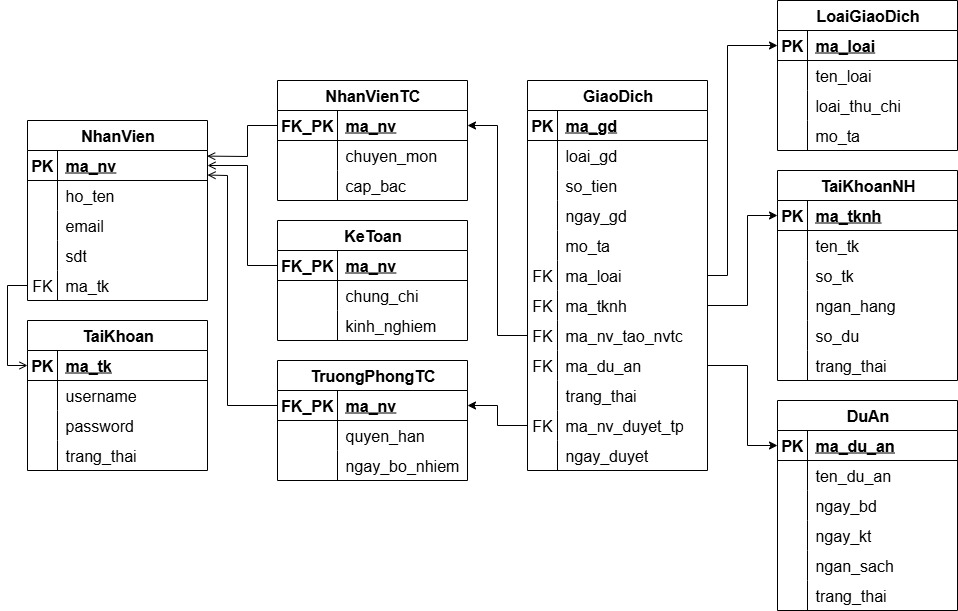
# **CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG**

## **1. Thiết kế cơ sở dữ liệu mức quan niệm**

Từ mô tả về dữ liệu cần có ở phần mô tả của bài toán ta hình thành được sơ đồ thực thể kết hợp (ERD).



## **2. Thiết kế cơ sở dữ liệu mức logic**



TaiKhoan (ma\_tk, username, password, trang\_thai)

NhanVien (ma\_nv, ho\_ten, email, sdt, ma\_tk)

TruongPhongTC (ma\_nv, quyen\_han, ngay\_bo\_nhiem)

NhanVienTC (ma\_nv, chuyen\_mon, cap\_bac)

KeToan (ma\_nv, chung\_chi, kinh\_nghiem)

LoaiGiaoDich (ma\_loai, ten\_loai, loai\_thu\_chi, mo\_ta)

TaiKhoanNH (ma\_tknh, ten\_tk, so\_tk, ngan\_hang, so\_du, trang\_thai)

DuAn (ma\_du\_an, ten\_du\_an, ngay\_bd, ngay\_kt, ngan\_sach, trang\_thai)

GiaoDich (ma\_gd, loai\_gd, so\_tien, ngay\_gd, mo\_ta, ma\_loai, ma\_tknh, ma\_nv\_tao\_nvtc, ma\_du\_an, trang\_thai, ma\_nv\_duyet\_tp, ngay\_duyet)

## **3. Các ràng buộc cần có**

|  |  |  |
| --- | --- | --- |
| **STT** | **Bảng** | **Ràng buộc** |
| 1 | **TaiKhoan**  ma\_tk INT IDENTITY,  username VARCHAR(50),  password VARCHAR(255),  trang\_thai VARCHAR(20) | Khóa chính ma\_tk  UNIQUE, NOT NULL username  NOT NULL password  CHECK: trang\_thai IN ('active', 'inactive') |
| 2 | **NhanVien**  ma\_nv INT IDENTITY,  ho\_ten NVARCHAR(100),  email NVARCHAR(100),  sdt VARCHAR(20),  ma\_tk INT | Khóa chính ma\_nv  Khóa ngoại ma\_tk tham chiếu đến bảng TaiKhoan  NOT NULL ho\_ten  UNIQUE, NOT NULL email  UNIQUE, NOT NULL ma\_tk |
| 3 | **TruongPhongTC**  ma\_nv INT,  quyen\_han NVARCHAR(200),  ngay\_bo\_nhiem DATE | Khóa chính ma\_nv  Khóa ngoại ma\_nv tham chiếu đến bảng NhanVien (ON DELETE CASCADE) |
| 4 | **NhanVienTC**  ma\_nv INT,  chuyen\_mon NVARCHAR(100),  cap\_bac VARCHAR(20) | Khóa chính ma\_nv  Khóa ngoại ma\_nv tham chiếu đến bảng NhanVien (ON DELETE CASCADE) |
| 5 | **KeToan**  ma\_nv INT,  chung\_chi NVARCHAR(200),  kinh\_nghiem INT | Khóa chính ma\_nv  Khóa ngoại ma\_nv tham chiếu đến bảng NhanVien (ON DELETE CASCADE)  CHECK: kinh\_nghiem >= 0 |
| 6 | **LoaiGiaoDich**  ma\_loai VARCHAR(10),  ten\_loai NVARCHAR(100),  loai\_thu\_chi VARCHAR(10),  mo\_ta NVARCHAR(200) | Khóa chính ma\_loai  NOT NULL ten\_loai  NOT NULL loai\_thu\_chi  CHECK: loai\_thu\_chi IN ('THU', 'CHI') |
| 7 | **TaiKhoanNH**  ma\_tknh INT IDENTITY,  ten\_tk NVARCHAR(100),  so\_tk VARCHAR(20),  ngan\_hang NVARCHAR(100),  so\_du DECIMAL(15,2),  trang\_thai VARCHAR(20) | Khóa chính ma\_tknh  NOT NULL ten\_tk  UNIQUE, NOT NULL so\_tk  NOT NULL ngan\_hang  CHECK: so\_du >= 0  CHECK: trang\_thai IN ('active', 'inactive') |
| 8 | **DuAn**  ma\_du\_an INT IDENTITY,  ten\_du\_an NVARCHAR(200),  ngay\_bd DATE,  ngay\_kt DATE NULL,  ngan\_sach DECIMAL(15,2),  trang\_thai VARCHAR(20) | Khóa chính ma\_du\_an  NOT NULL ten\_du\_an  NOT NULL ngay\_bd  CHECK: ngay\_kt IS NULL OR ngay\_kt >= ngay\_bd  CHECK: ngan\_sach >= 0  CHECK: trang\_thai IN ('active', 'inactive', 'hoan\_thanh')  CHECK: (trang\_thai <> 'hoan\_thanh' OR ngay\_kt IS NOT NULL) |
| 9 | **GiaoDich**  ma\_gd INT IDENTITY,  loai\_gd VARCHAR(10),  so\_tien DECIMAL(15,2),  ngay\_gd DATETIME,  mo\_ta NVARCHAR(500),  ma\_loai VARCHAR(10),  ma\_tknh INT,  ma\_nv\_tao\_nvtc INT,  ma\_du\_an INT NULL,  trang\_thai VARCHAR(20),  ma\_nv\_duyet\_tp INT NULL,  ngay\_duyet DATETIME NULL | Khóa chính ma\_gd  Khóa ngoại ma\_loai tham chiếu đến bảng LoaiGiaoDich  Khóa ngoại ma\_tknh tham chiếu đến bảng TaiKhoanNH  Khóa ngoại ma\_nv\_tao\_nvtc tham chiếu đến bảng NhanVienTC  Khóa ngoại ma\_du\_an tham chiếu đến bảng DuAn  Khóa ngoại ma\_nv\_duyet\_tp tham chiếu đến bảng TruongPhongTC  NOT NULL loai\_gd  NOT NULL so\_tien  NOT NULL ma\_loai  NOT NULL ma\_tknh  NOT NULL ma\_nv\_tao\_nvtc  CHECK: loai\_gd IN ('THU', 'CHI')  CHECK: so\_tien > 0  CHECK: trang\_thai IN ('CHO\_DUYET', 'DA\_DUYET', 'TU\_CHOI')  CHECK duyệt hợp lệ: nếu trang\_thai IN ('DA\_DUYET', 'TU\_CHOI') thì ma\_nv\_duyet\_tp, ngay\_duyet NOT NULL và ngay\_duyet ≥ ngay\_gd |

## **4. Cài đặt SQL và các ràng buộc**

**Bảng Tài khoản**

CREATE TABLE TaiKhoan (

ma\_tk INT IDENTITY(1,1) PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL,

trang\_thai VARCHAR(20) DEFAULT 'active'

);

**Bảng Nhân viên**

CREATE TABLE NhanVien (

ma\_nv INT IDENTITY(1,1) PRIMARY KEY,

ho\_ten NVARCHAR(100) NOT NULL,

email NVARCHAR(100) UNIQUE NOT NULL,

sdt VARCHAR(20),

ma\_tk INT UNIQUE NOT NULL,

FOREIGN KEY (ma\_tk) REFERENCES TaiKhoan(ma\_tk)

);

**Bảng Trưởng phòng tài chính**

CREATE TABLE TruongPhongTC (

ma\_nv INT PRIMARY KEY,

quyen\_han NVARCHAR(200),

ngay\_bo\_nhiem DATE DEFAULT CAST(GETDATE() AS DATE),

FOREIGN KEY (ma\_nv) REFERENCES NhanVien(ma\_nv) ON DELETE CASCADE

);

**Bảng Nhân viên tài chính**

CREATE TABLE NhanVienTC (

ma\_nv INT PRIMARY KEY,

chuyen\_mon NVARCHAR(100),

cap\_bac VARCHAR(20) DEFAULT 'NhanVien',

FOREIGN KEY (ma\_nv) REFERENCES NhanVien(ma\_nv) ON DELETE CASCADE

);

**Bảng Kế toán**

CREATE TABLE KeToan (

ma\_nv INT PRIMARY KEY,

chung\_chi NVARCHAR(200),

kinh\_nghiem INT DEFAULT 0,

FOREIGN KEY (ma\_nv) REFERENCES NhanVien(ma\_nv) ON DELETE CASCADE

);

**Bảng Loại giao dịch**

CREATE TABLE LoaiGiaoDich (

ma\_loai VARCHAR(10) PRIMARY KEY,

ten\_loai NVARCHAR(100) NOT NULL,

loai\_thu\_chi VARCHAR(10) NOT NULL,

mo\_ta NVARCHAR(200)

);

**Bảng Tài khoản ngân hàng**

CREATE TABLE TaiKhoanNH (

ma\_tknh INT IDENTITY(1,1) PRIMARY KEY,

ten\_tk NVARCHAR(100) NOT NULL,

so\_tk VARCHAR(20) UNIQUE NOT NULL,

ngan\_hang NVARCHAR(100) NOT NULL,

so\_du DECIMAL(15,2) DEFAULT 0,

trang\_thai VARCHAR(20) DEFAULT 'active'

);

**Bảng Dự án**

CREATE TABLE DuAn (

ma\_du\_an INT IDENTITY(1,1) PRIMARY KEY,

ten\_du\_an NVARCHAR(200) NOT NULL,

ngay\_bd DATE NOT NULL,

ngay\_kt DATE,

ngan\_sach DECIMAL(15,2) DEFAULT 0,

trang\_thai VARCHAR(20) DEFAULT 'active'

);

**Bảng Giao dịch**

CREATE TABLE GiaoDich (

ma\_gd INT IDENTITY(1,1) PRIMARY KEY,

loai\_gd VARCHAR(10) NOT NULL,

so\_tien DECIMAL(15,2) NOT NULL,

ngay\_gd DATETIME DEFAULT GETDATE(),

mo\_ta NVARCHAR(500),

ma\_loai VARCHAR(10) NOT NULL,

ma\_tknh INT NOT NULL,

ma\_nv\_tao\_nvtc INT NOT NULL,

ma\_du\_an INT,

trang\_thai VARCHAR(20) DEFAULT 'CHO\_DUYET',

ma\_nv\_duyet\_tp INT,

ngay\_duyet DATETIME,

FOREIGN KEY (ma\_loai) REFERENCES LoaiGiaoDich(ma\_loai),

FOREIGN KEY (ma\_tknh) REFERENCES TaiKhoanNH(ma\_tknh),

FOREIGN KEY (ma\_nv\_tao\_nvtc) REFERENCES NhanVienTC(ma\_nv),

FOREIGN KEY (ma\_du\_an) REFERENCES DuAn(ma\_du\_an),

FOREIGN KEY (ma\_nv\_duyet\_tp) REFERENCES TruongPhongTC(ma\_nv)

);

**Các ràng buộc liên quan khác**

ALTER TABLE TaiKhoan ADD CONSTRAINT CK\_TaiKhoan\_TrangThai

CHECK (trang\_thai IN ('active', 'inactive'));

ALTER TABLE KeToan ADD CONSTRAINT CK\_KeToan\_KinhNghiem

CHECK (kinh\_nghiem >= 0);

ALTER TABLE LoaiGiaoDich ADD CONSTRAINT CK\_LoaiGiaoDich\_LoaiThuChi

CHECK (loai\_thu\_chi IN ('THU', 'CHI'));

ALTER TABLE TaiKhoanNH ADD CONSTRAINT CK\_TaiKhoanNH\_TrangThai

CHECK (trang\_thai IN ('active', 'inactive'));

ALTER TABLE TaiKhoanNH ADD CONSTRAINT CK\_TaiKhoanNH\_SoDu

CHECK (so\_du >= 0);

ALTER TABLE DuAn ADD CONSTRAINT CK\_DuAn\_TrangThai

CHECK (trang\_thai IN ('active', 'inactive', 'hoan\_thanh'));

ALTER TABLE DuAn ADD CONSTRAINT CK\_DuAn\_NganSach

CHECK (ngan\_sach >= 0);

ALTER TABLE DuAn ADD CONSTRAINT CK\_DuAn\_Ngay

CHECK (ngay\_kt IS NULL OR ngay\_kt >= ngay\_bd);

ALTER TABLE DuAn ADD CONSTRAINT CK\_DuAn\_HoanThanhCoNgayKt

CHECK (trang\_thai <> 'hoan\_thanh' OR ngay\_kt IS NOT NULL);

ALTER TABLE GiaoDich ADD CONSTRAINT CK\_GiaoDich\_LoaiGd

CHECK (loai\_gd IN ('THU', 'CHI'));

ALTER TABLE GiaoDich ADD CONSTRAINT CK\_GiaoDich\_TrangThai

CHECK (trang\_thai IN ('CHO\_DUYET', 'DA\_DUYET', 'TU\_CHOI'));

ALTER TABLE GiaoDich ADD CONSTRAINT CK\_GiaoDich\_SoTien

CHECK (so\_tien > 0);

ALTER TABLE GiaoDich ADD CONSTRAINT CK\_GiaoDich\_DuyetHopLe

CHECK (

(trang\_thai='CHO\_DUYET')

OR (

trang\_thai IN ('DA\_DUYET','TU\_CHOI')

AND ma\_nv\_duyet\_tp IS NOT NULL

AND ngay\_duyet IS NOT NULL

AND ngay\_duyet >= ngay\_gd

)

);

## **5. Lược đồ quan hệ sau khi cài đặt SQL**

## 

## **6. Các view**

**a. View xem danh sách nhân viên**

CREATE OR ALTER VIEW dbo.V\_DanhSachNhanVien

AS

SELECT

nv.ma\_nv,

nv.ho\_ten,

nv.email,

nv.sdt,

tk.username,

tk.trang\_thai,

CAST(CASE

WHEN EXISTS (SELECT 1 FROM dbo.TruongPhongTC tp WHERE tp.ma\_nv = nv.ma\_nv) THEN N'TRUONG\_PHONG\_TC'

WHEN EXISTS (SELECT 1 FROM dbo.NhanVienTC nvtc WHERE nvtc.ma\_nv = nv.ma\_nv) THEN N'NHAN\_VIEN\_TC'

WHEN EXISTS (SELECT 1 FROM dbo.KeToan kt WHERE kt.ma\_nv = nv.ma\_nv) THEN N'KE\_TOAN'

ELSE N'UNKNOWN'

END AS NVARCHAR(30)) AS vai\_tro

FROM dbo.NhanVien nv

JOIN dbo.TaiKhoan tk ON tk.ma\_tk = nv.ma\_tk;

**b. View xem thông tin dự án**

CREATE VIEW V\_DuAn

AS

SELECT ma\_du\_an, ten\_du\_an, ngay\_bd, ngay\_kt, ngan\_sach, trang\_thai

FROM DuAn;

**c. View xem thông tin tài khoản ngân hàng**

CREATE VIEW V\_TaiKhoanNH

AS

SELECT ma\_tknh, ten\_tk, so\_tk, ngan\_hang, so\_du, trang\_thai

FROM TaiKhoanNH;

**d. View xem thông tin các loại giao dịch**

CREATE VIEW V\_LoaiGiaoDich

AS

SELECT ma\_loai, ten\_loai, loai\_thu\_chi, mo\_ta

FROM LoaiGiaoDich;

**e. View xem thông tin chi tiết các giao dịch chờ duyệt**

CREATE VIEW V\_GiaoDichChoDuyet

AS

SELECT

gd.ma\_gd,

gd.loai\_gd,

gd.so\_tien,

gd.ngay\_gd,

gd.mo\_ta,

lg.ten\_loai,

nv\_tao.ho\_ten as nguoi\_tao,

da.ten\_du\_an

FROM GiaoDich gd

INNER JOIN LoaiGiaoDich lg ON gd.ma\_loai = lg.ma\_loai

INNER JOIN NhanVienTC nvtc ON gd.ma\_nv\_tao\_nvtc = nvtc.ma\_nv

INNER JOIN NhanVien nv\_tao ON nvtc.ma\_nv = nv\_tao.ma\_nv

LEFT JOIN DuAn da ON gd.ma\_du\_an = da.ma\_du\_an

WHERE gd.trang\_thai = 'CHO\_DUYET' -- Chỉ lấy gd chờ duyệt

**f. View xem các thông tin chi tiết về lịch sử giao dịch**

CREATE VIEW V\_LichSuGiaoDich

AS

SELECT

gd.ma\_gd,

gd.loai\_gd,

gd.so\_tien,

gd.ngay\_gd,

gd.mo\_ta,

gd.trang\_thai,

lg.ten\_loai,

tkn.ten\_tk as tai\_khoan,

nv\_tao.ho\_ten as nguoi\_tao,

nv\_duyet\_tp.ho\_ten as nguoi\_duyet,

gd.ngay\_duyet,

da.ten\_du\_an

FROM GiaoDich gd

INNER JOIN LoaiGiaoDich lg ON gd.ma\_loai = lg.ma\_loai

INNER JOIN TaiKhoanNH tkn ON gd.ma\_tknh = tkn.ma\_tknh

INNER JOIN NhanVienTC nvtc ON gd.ma\_nv\_tao\_nvtc = nvtc.ma\_nv

INNER JOIN NhanVien nv\_tao ON nvtc.ma\_nv = nv\_tao.ma\_nv

LEFT JOIN TruongPhongTC tp ON gd.ma\_nv\_duyet\_tp = tp.ma\_nv

LEFT JOIN NhanVien nv\_duyet\_tp ON tp.ma\_nv = nv\_duyet\_tp.ma\_nv

LEFT JOIN DuAn da ON gd.ma\_du\_an = da.ma\_du\_an

**g. View xem thống kê tài chính của một tháng trong năm**

CREATE VIEW V\_ThongKeThang

AS

SELECT

YEAR(ngay\_gd) as Nam,

MONTH(ngay\_gd) as Thang,

SUM(CASE WHEN loai\_gd = 'THU' THEN so\_tien ELSE 0 END) as TongThu,

SUM(CASE WHEN loai\_gd = 'CHI' THEN so\_tien ELSE 0 END) as TongChi,

SUM(CASE WHEN loai\_gd = 'THU' THEN so\_tien ELSE -so\_tien END) as LaiLo

FROM GiaoDich

WHERE trang\_thai = 'DA\_DUYET'

GROUP BY YEAR(ngay\_gd), MONTH(ngay\_gd)

## **7. Các trigger**

**a. Trigger tự động cập nhật số dư tài khoản sau khi duyệt giao dịch**

CREATE OR ALTER TRIGGER TR\_GiaoDich\_UpdateSoDu

ON dbo.GiaoDich

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

DECLARE @Transitions TABLE (

ma\_tknh INT PRIMARY KEY,

delta DECIMAL(15,2)

);

INSERT INTO @Transitions(ma\_tknh, delta)

SELECT i.ma\_tknh,

SUM(CASE WHEN i.loai\_gd='THU' THEN i.so\_tien

WHEN i.loai\_gd='CHI' THEN -i.so\_tien ELSE 0 END) AS delta

FROM inserted i

JOIN deleted d ON d.ma\_gd = i.ma\_gd

WHERE d.trang\_thai = 'CHO\_DUYET' AND i.trang\_thai = 'DA\_DUYET'

GROUP BY i.ma\_tknh;

-- Chặn âm số dư

IF EXISTS (

SELECT 1

FROM dbo.TaiKhoanNH t

JOIN @Transitions tr ON tr.ma\_tknh = t.ma\_tknh

WHERE t.so\_du + tr.delta < 0

)

BEGIN

RAISERROR(N'Số dư sẽ âm, không thể duyệt giao dịch.', 16, 1);

RETURN;

END

-- Cập nhật số dư

UPDATE t

SET t.so\_du = t.so\_du + tr.delta

FROM dbo.TaiKhoanNH t

JOIN @Transitions tr ON tr.ma\_tknh = t.ma\_tknh;

END;

**b. Trigger chặn thao tác xóa giao dịch trên bảng GiaoDich**

CREATE OR ALTER TRIGGER TR\_GiaoDich\_PreventDelete

ON dbo.GiaoDich

INSTEAD OF DELETE

AS

BEGIN

SET NOCOUNT ON;

THROW 51001, N'Không được phép xóa giao dịch! Chỉ có thể cập nhật trạng thái hoặc từ chối giao dịch.', 1;

END;

**c. Trigger vô hiệu hóa tài khoản nhân viên khi dùng lệnh xóa trên TaiKhoan**

CREATE OR ALTER TRIGGER TR\_TaiKhoan\_InsteadOfDelete

ON dbo.TaiKhoan

INSTEAD OF DELETE

AS

BEGIN

SET NOCOUNT ON;

UPDATE tk

SET tk.trang\_thai = 'inactive'

FROM dbo.TaiKhoan AS tk

INNER JOIN deleted AS d ON d.ma\_tk = tk.ma\_tk;

END;

**d. Trigger vô hiệu hóa tài khoản ngân hàng khi dùng lệnh xóa trên TaiKhoanNH**

CREATE OR ALTER TRIGGER TR\_TaiKhoanNH\_InsteadOfDelete

ON dbo.TaiKhoanNH

INSTEAD OF DELETE

AS

BEGIN

SET NOCOUNT ON;

UPDATE tknh

SET tknh.trang\_thai = 'inactive'

FROM dbo.TaiKhoanNH AS tknh

INNER JOIN deleted AS d ON d.ma\_tknh = tknh.ma\_tknh;

END;

**e. Trigger vô hiệu hóa dự án khi thực hiện lệnh xóa trên DuAn**

CREATE OR ALTER TRIGGER TR\_DuAn\_InsteadOfDelete

ON dbo.DuAn

INSTEAD OF DELETE

AS

BEGIN

SET NOCOUNT ON;

UPDATE da

SET da.trang\_thai = 'inactive'

FROM dbo.DuAn AS da

INNER JOIN deleted AS d ON d.ma\_du\_an = da.ma\_du\_an;

END;

**f. Trigger ngăn chặn sự chuyển đổi trạng thái không hợp lệ trên bảng GiaoDich**

CREATE OR ALTER TRIGGER TR\_GiaoDich\_ValidateTransition

ON dbo.GiaoDich

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

-- Chặn chuyển trạng thái từ DA\_DUYET/TU\_CHOI sang bất kỳ trạng thái khác

IF EXISTS (

SELECT 1

FROM inserted i

JOIN deleted d ON d.ma\_gd = i.ma\_gd

WHERE d.trang\_thai IN ('DA\_DUYET','TU\_CHOI')

AND i.trang\_thai <> d.trang\_thai

)

BEGIN

RAISERROR(N'Không được phép thay đổi trạng thái sau khi đã duyệt hoặc từ chối.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

-- Với trạng thái ban đầu là CHO\_DUYET, chỉ cho phép sang DA\_DUYET hoặc TU\_CHOI (hoặc giữ nguyên)

IF EXISTS (

SELECT 1

FROM inserted i

JOIN deleted d ON d.ma\_gd = i.ma\_gd

WHERE d.trang\_thai = 'CHO\_DUYET'

AND i.trang\_thai NOT IN ('CHO\_DUYET','DA\_DUYET','TU\_CHOI')

)

BEGIN

RAISERROR(N'Trạng thái không hợp lệ. CHO\_DUYET chỉ được chuyển sang DA\_DUYET hoặc TU\_CHOI.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

END;

**g. Trigger ngăn chặn việc thực hiện giao dịch trên các tài khoản ngân hàng hoặc**

**các dự án không hoạt động**

CREATE OR ALTER TRIGGER TR\_GiaoDich\_ValidateFKActive

ON dbo.GiaoDich

AFTER INSERT, UPDATE

AS

BEGIN

SET NOCOUNT ON;

-- Tài khoản ngân hàng phải còn hoạt động

IF EXISTS (

SELECT 1

FROM inserted i

JOIN dbo.TaiKhoanNH tknh ON tknh.ma\_tknh = i.ma\_tknh

WHERE tknh.trang\_thai <> 'active'

)

BEGIN

RAISERROR(N'Tài khoản ngân hàng đã inactive. Không thể tạo/cập nhật giao dịch.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

-- Nếu có dự án, dự án phải còn hoạt động

IF EXISTS (

SELECT 1

FROM inserted i

JOIN dbo.DuAn da ON da.ma\_du\_an = i.ma\_du\_an

WHERE i.ma\_du\_an IS NOT NULL

AND da.trang\_thai <> 'active'

)

BEGIN

RAISERROR(N'Dự án đã inactive/hoàn thành. Không thể tạo/cập nhật giao dịch gắn dự án này.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

END;

# **CHƯƠNG 3: THIẾT KẾ CÁC CHỨC NĂNG**

## **1. Kết nối cơ sở dữ liệu**

public static class DatabaseConnection

{

private const string Server = @"WIN";

private const string Database = "JobCenterFinancialManagement";

private static string connectionString;

public static void Configure(string user, string password)

{

var stringbuilder = new SqlConnectionStringBuilder

{

DataSource = Server,

InitialCatalog = Database,

UserID = user?.Trim(),

Password = password?.Trim(),

MultipleActiveResultSets = true,

Encrypt = false,

TrustServerCertificate = true

};

connectionString = stringbuilder.ConnectionString;

}

public static SqlConnection CreateConnection()

{

if (string.IsNullOrWhiteSpace(connectionString))

throw new InvalidOperationException("Chưa gọi đúng database");

return new SqlConnection(connectionString);

}

public static void CloseConnection(SqlConnection conn)

{

if (conn != null && conn.State == ConnectionState.Open)

conn.Close();

}

}

public static class DbHelper

{

public static SqlCommand StoredProc(string spName)

{

var cmd = new SqlCommand(spName) { CommandType = CommandType.StoredProcedure };

return cmd;

}

public static SqlParameter Param(string name, object value)

{

return new SqlParameter(name, value ?? DBNull.Value);

}

public static DataTable ExecuteDataTable(SqlCommand cmd)

{

try

{

using (var conn = DatabaseConnection.CreateConnection())

using (var da = new SqlDataAdapter())

{

cmd.Connection = conn;

var dt = new DataTable();

da.SelectCommand = cmd;

da.Fill(dt);

return dt;

}

}

catch (Exception ex)

{

throw new DataAccessException($"Lỗi {cmd.CommandText}", ex);

}

}

public static object ExecuteScalar(SqlCommand cmd)

{

try

{

using (var conn = DatabaseConnection.CreateConnection())

{

cmd.Connection = conn;

conn.Open();

return cmd.ExecuteScalar();

}

}

catch (Exception ex)

{

throw new DataAccessException($"Lỗi {cmd.CommandText}", ex);

}

}

public static int ExecuteNonQuery(SqlCommand cmd)

{

try

{

using (var conn = DatabaseConnection.CreateConnection())

{

cmd.Connection = conn;

conn.Open();

return cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

throw new DataAccessException($"Lỗi {cmd.CommandText}", ex);

}

}

}

## **2. Chức năng riêng của nhân viên tài chính**

**a. Xem lịch sử giao dịch mà nhân viên tài chính tạo**

CREATE OR ALTER PROCEDURE dbo.SP\_GetLichSuGiaoDich\_ForNhanVienTC

@MaNvTao INT,

@TuNgay DATETIME = NULL,

@DenNgay DATETIME = NULL,

@TrangThai VARCHAR(20) = NULL,

@LoaiGd VARCHAR(10) = NULL,

@MaDuAn INT = NULL,

@MaTknh INT = NULL

AS

BEGIN

SET NOCOUNT ON;

-- Kiểm tra quyền: phải là nhân viên TC

IF NOT EXISTS (SELECT 1 FROM dbo.NhanVienTC WHERE ma\_nv = @MaNvTao)

BEGIN

RAISERROR(N'Không có quyền xem lịch sử giao dịch!', 16, 1);

RETURN;

END;

SELECT

v.ma\_gd,

v.loai\_gd,

v.so\_tien,

v.ngay\_gd,

v.mo\_ta,

v.trang\_thai,

v.ten\_loai,

v.tai\_khoan,

v.nguoi\_tao,

v.nguoi\_duyet,

v.ngay\_duyet,

v.ten\_du\_an

FROM dbo.V\_LichSuGiaoDich AS v

INNER JOIN dbo.GiaoDich AS gd

ON gd.ma\_gd = v.ma\_gd

WHERE gd.ma\_nv\_tao\_nvtc = @MaNvTao

AND (@TuNgay IS NULL OR v.ngay\_gd >= @TuNgay)

AND (@DenNgay IS NULL OR v.ngay\_gd < DATEADD(DAY, 1, @DenNgay))

AND (@TrangThai IS NULL OR v.trang\_thai = @TrangThai)

AND (@LoaiGd IS NULL OR v.loai\_gd = @LoaiGd)

AND (@MaDuAn IS NULL OR gd.ma\_du\_an = @MaDuAn)

AND (@MaTknh IS NULL OR gd.ma\_tknh = @MaTknh)

ORDER BY v.ngay\_gd DESC;

END

**Xử lý bên C#:**

public DataTable GetLichSuGiaoDich\_CuaToi(int maNvTao, DateTime? tuNgay = null, DateTime? denNgay = null, string trangThai = null, string loaiGd = null, int? maDuAn = null, int? maTknh = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_GetLichSuGiaoDich\_ForNhanVienTC");

cmd.Parameters.Add(DbHelper.Param("@MaNvTao", maNvTao));

cmd.Parameters.Add(DbHelper.Param("@TuNgay", tuNgay));

cmd.Parameters.Add(DbHelper.Param("@DenNgay", denNgay));

cmd.Parameters.Add(DbHelper.Param("@TrangThai", trangThai));

cmd.Parameters.Add(DbHelper.Param("@LoaiGd", loaiGd));

cmd.Parameters.Add(DbHelper.Param("@MaDuAn", maDuAn));

cmd.Parameters.Add(DbHelper.Param("@MaTknh", maTknh));

return DbHelper.ExecuteDataTable(cmd);

}

**b. Tạo giao dịch mới**

CREATE PROCEDURE SP\_ThemGiaoDich

@loai\_gd VARCHAR(10),

@so\_tien DECIMAL(15,2),

@mo\_ta NVARCHAR(500),

@ma\_loai VARCHAR(10),

@ma\_tknh INT,

@ma\_nv\_tao\_nvtc INT,

@ma\_du\_an INT = NULL

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra loai\_gd khớp với loai\_thu\_chi của mã loại

IF NOT EXISTS (

SELECT 1

FROM LoaiGiaoDich lgd

WHERE lgd.ma\_loai = @ma\_loai

AND lgd.loai\_thu\_chi = @loai\_gd

)

BEGIN

RAISERROR(N'Loại giao dịch không khớp với mã loại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Thêm giao dịch mới

INSERT INTO GiaoDich (loai\_gd, so\_tien, mo\_ta, ma\_loai, ma\_tknh, ma\_nv\_tao\_nvtc, ma\_du\_an)

VALUES (@loai\_gd, @so\_tien, @mo\_ta, @ma\_loai, @ma\_tknh, @ma\_nv\_tao\_nvtc, @ma\_du\_an);

SELECT SCOPE\_IDENTITY() as ma\_gd\_moi;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int ThemGiaoDich(string loaiGd, decimal soTien, string moTa, string maLoai, int maTknh, int maNvTaoNvtc, int? maDuAn)

{

var cmd = DbHelper.StoredProc("dbo.SP\_ThemGiaoDich");

cmd.Parameters.Add(DbHelper.Param("@loai\_gd", loaiGd));

cmd.Parameters.Add(DbHelper.Param("@so\_tien", soTien));

cmd.Parameters.Add(DbHelper.Param("@mo\_ta", moTa));

cmd.Parameters.Add(DbHelper.Param("@ma\_loai", maLoai));

cmd.Parameters.Add(DbHelper.Param("@ma\_tknh", maTknh));

cmd.Parameters.Add(DbHelper.Param("@ma\_nv\_tao\_nvtc", maNvTaoNvtc));

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**c. Sửa giao dịch đang ở trạng thái chờ duyệt**

CREATE PROCEDURE SP\_SuaGiaoDich

@ma\_gd INT,

@so\_tien DECIMAL(15,2),

@mo\_ta NVARCHAR(500),

@ma\_loai VARCHAR(10),

@ma\_tknh INT,

@ma\_du\_an INT = NULL,

@ma\_nv\_sua INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra giao dịch có tồn tại và chưa duyệt không

IF NOT EXISTS (SELECT 1 FROM GiaoDich WHERE ma\_gd = @ma\_gd AND trang\_thai = 'CHO\_DUYET')

BEGIN

RAISERROR(N'Giao dịch không tồn tại hoặc đã được duyệt!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- KIỂM TRA QUYỀN: Chỉ được sửa giao dịch của mình

IF NOT EXISTS (SELECT 1 FROM GiaoDich WHERE ma\_gd = @ma\_gd AND ma\_nv\_tao\_nvtc = @ma\_nv\_sua)

BEGIN

RAISERROR(N'Không có quyền sửa giao dịch của người khác!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra loai\_gd hiện tại khớp với loai\_thu\_chi của mã loại mới

DECLARE @loai\_gd\_hien\_tai VARCHAR(10);

SELECT @loai\_gd\_hien\_tai = loai\_gd FROM GiaoDich WHERE ma\_gd = @ma\_gd;

IF NOT EXISTS (

SELECT 1

FROM LoaiGiaoDich lgd

WHERE lgd.ma\_loai = @ma\_loai

AND lgd.loai\_thu\_chi = @loai\_gd\_hien\_tai

)

BEGIN

RAISERROR(N'Loại giao dịch không khớp THU/CHI với mã loại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật giao dịch

UPDATE GiaoDich

SET so\_tien = @so\_tien,

mo\_ta = @mo\_ta,

ma\_loai = @ma\_loai,

ma\_tknh = @ma\_tknh,

ma\_du\_an = @ma\_du\_an

WHERE ma\_gd = @ma\_gd;

-- Trả về số dòng được cập nhật

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int SuaGiaoDich(int maGd, decimal soTien, string moTa, string maLoai, int maTknh, int? maDuAn, int maNvSua)

{

var cmd = DbHelper.StoredProc("dbo.SP\_SuaGiaoDich");

cmd.Parameters.Add(DbHelper.Param("@ma\_gd", maGd));

cmd.Parameters.Add(DbHelper.Param("@so\_tien", soTien));

cmd.Parameters.Add(DbHelper.Param("@mo\_ta", moTa));

cmd.Parameters.Add(DbHelper.Param("@ma\_loai", maLoai));

cmd.Parameters.Add(DbHelper.Param("@ma\_tknh", maTknh));

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

cmd.Parameters.Add(DbHelper.Param("@ma\_nv\_sua", maNvSua));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**d. Kiểm tra số dư trước khi tạo hoặc sửa giao dịch**

CREATE FUNCTION FN\_KiemTraSoDu(@ma\_tknh INT)

RETURNS DECIMAL(15,2)

AS

BEGIN

DECLARE @so\_du DECIMAL(15,2);

-- Lấy số dư hiện tại của tài khoản

SELECT @so\_du = so\_du

FROM TaiKhoanNH

WHERE ma\_tknh = @ma\_tknh;

-- Trả về số dư (0 nếu không tìm thấy)

RETURN ISNULL(@so\_du, 0);

END;

**Xử lý bên C#:**

public decimal KiemTraSoDu(int maTknh)

{

using (var cmd = new SqlCommand(

"SELECT dbo.FN\_KiemTraSoDu(@ma\_tknh);")

{ CommandType = CommandType.Text })

{

cmd.Parameters.Add(DbHelper.Param("@ma\_tknh", maTknh));

var scalar = DbHelper.ExecuteScalar(cmd);

return scalar == null || scalar == DBNull.Value ? 0m : Convert.ToDecimal(scalar);

}

}

## **3. Chức năng riêng của trưởng phòng tài chính**

**a. Xem thông tin chi tiết các giao dịch đang chờ duyệt**

CREATE OR ALTER PROCEDURE dbo.SP\_GetGiaoDichChoDuyet

@MaNvTao INT = NULL,

@LoaiGd VARCHAR(10) = NULL,

@MaDuAn INT = NULL

AS

BEGIN

SET NOCOUNT ON;

SELECT

v.ma\_gd,

v.loai\_gd,

v.so\_tien,

v.ngay\_gd,

v.mo\_ta,

v.ten\_loai,

v.nguoi\_tao,

v.ten\_du\_an

FROM dbo.V\_GiaoDichChoDuyet AS v

INNER JOIN dbo.GiaoDich AS gd

ON gd.ma\_gd = v.ma\_gd

WHERE (@MaNvTao IS NULL OR gd.ma\_nv\_tao\_nvtc = @MaNvTao)

AND (@LoaiGd IS NULL OR v.loai\_gd = @LoaiGd)

AND (@MaDuAn IS NULL OR gd.ma\_du\_an = @MaDuAn)

ORDER BY v.ngay\_gd DESC;

END

**Xử lý bên C#:**

public DataTable GetGiaoDichChoDuyet(int? maNvTao = null, string loaiGd = null, int? maDuAn = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_GetGiaoDichChoDuyet");

cmd.Parameters.Add(DbHelper.Param("@MaNvTao", maNvTao));

cmd.Parameters.Add(DbHelper.Param("@LoaiGd", loaiGd));

cmd.Parameters.Add(DbHelper.Param("@MaDuAn", maDuAn));

return DbHelper.ExecuteDataTable(cmd);

}

**b. Duyệt hoặc từ chối các giao dịch được nhân viên tài chính tạo**

CREATE PROCEDURE SP\_DuyetGiaoDich

@ma\_gd INT,

@ma\_nv\_duyet\_tp INT,

@trang\_thai VARCHAR(20)

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra giao dịch có tồn tại và đang chờ duyệt không

IF NOT EXISTS (SELECT 1 FROM GiaoDich WHERE ma\_gd = @ma\_gd AND trang\_thai = 'CHO\_DUYET')

BEGIN

RAISERROR(N'Giao dịch không tồn tại hoặc không ở trạng thái chờ duyệt!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra quyền duyệt (chỉ Trưởng phòng mới được duyệt)

IF NOT EXISTS (

SELECT 1 FROM TruongPhongTC tp

WHERE tp.ma\_nv = @ma\_nv\_duyet\_tp

)

BEGIN

RAISERROR(N'Không có quyền duyệt giao dịch!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra số dư nếu là giao dịch chi tại thời điểm duyệt

DECLARE @loai\_gd\_cur VARCHAR(10);

DECLARE @ma\_tknh\_cur INT;

DECLARE @so\_tien\_cur DECIMAL(15,2);

SELECT @loai\_gd\_cur = loai\_gd, @ma\_tknh\_cur = ma\_tknh, @so\_tien\_cur = so\_tien

FROM GiaoDich WHERE ma\_gd = @ma\_gd;

IF @trang\_thai = 'DA\_DUYET' AND @loai\_gd\_cur = 'CHI'

BEGIN

DECLARE @so\_du\_hien\_tai DECIMAL(15,2);

SELECT @so\_du\_hien\_tai = so\_du FROM TaiKhoanNH WHERE ma\_tknh = @ma\_tknh\_cur;

IF @so\_du\_hien\_tai < @so\_tien\_cur

BEGIN

RAISERROR(N'Số dư tài khoản không đủ để duyệt giao dịch chi!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

END

-- Cập nhật giao dịch với thông tin duyệt

UPDATE GiaoDich

SET trang\_thai = @trang\_thai,

ma\_nv\_duyet\_tp = @ma\_nv\_duyet\_tp,

ngay\_duyet = GETDATE()

WHERE ma\_gd = @ma\_gd;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public bool DuyetGiaoDich(int maGd, int maNvDuyetTp, string trangThai)

{

var cmd = DbHelper.StoredProc("dbo.SP\_DuyetGiaoDich");

cmd.Parameters.Add(DbHelper.Param("@ma\_gd", maGd));

cmd.Parameters.Add(DbHelper.Param("@ma\_nv\_duyet\_tp", maNvDuyetTp));

cmd.Parameters.Add(DbHelper.Param("@trang\_thai", trangThai));

DbHelper.ExecuteNonQuery(cmd);

return true;

}

**c. Tìm kiếm nhân viên dựa trên thông tin cá nhân và tài khoản**

CREATE OR ALTER PROCEDURE dbo.SP\_TimNhanVien

@Keyword NVARCHAR(100) = NULL,

@VaiTro NVARCHAR(30) = NULL,

@TrangThai VARCHAR(20) = NULL

AS

BEGIN

SET NOCOUNT ON;

SELECT ma\_nv, ho\_ten, email, sdt, username, trang\_thai, vai\_tro

FROM dbo.V\_DanhSachNhanVien

WHERE (@TrangThai IS NULL OR trang\_thai = @TrangThai)

AND (@VaiTro IS NULL OR vai\_tro = @VaiTro)

AND (

@Keyword IS NULL

OR ho\_ten LIKE N'%' + @Keyword + N'%'

OR email LIKE N'%' + @Keyword + N'%'

OR username LIKE N'%' + @Keyword + N'%'

)

ORDER BY ho\_ten ASC;

END;

**Xử lý bên C#:**

public DataTable TimNhanVien(string keyword = null, string vaiTro = null, string trangThai = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_TimNhanVien");

cmd.Parameters.Add(DbHelper.Param("@Keyword", keyword));

cmd.Parameters.Add(DbHelper.Param("@VaiTro", vaiTro));

cmd.Parameters.Add(DbHelper.Param("@TrangThai", trangThai));

return DbHelper.ExecuteDataTable(cmd);

}

**d. Thêm thông tin nhân viên và tạo tài khoản cho nhân viên đó**

CREATE OR ALTER PROCEDURE dbo.SP\_ThemNhanVien

@HoTen NVARCHAR(100),

@Email NVARCHAR(100),

@Sdt VARCHAR(20) = NULL,

@Username VARCHAR(50),

@Password VARCHAR(255),

@VaiTro NVARCHAR(30),

@Provision BIT = 1

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRAN;

BEGIN TRY

IF EXISTS (SELECT 1 FROM dbo.TaiKhoan WHERE username = @Username)

BEGIN

RAISERROR(N'Username đã tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

IF EXISTS (SELECT 1 FROM dbo.NhanVien WHERE email = @Email)

BEGIN

RAISERROR(N'Email đã tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

-- 1) Tài khoản ứng dụng

INSERT INTO dbo.TaiKhoan(username, [password], trang\_thai)

VALUES (@Username, @Password, 'active');

DECLARE @ma\_tk INT = SCOPE\_IDENTITY();

-- 2) Nhân viên

INSERT INTO dbo.NhanVien(ho\_ten, email, sdt, ma\_tk)

VALUES (@HoTen, @Email, @Sdt, @ma\_tk);

DECLARE @ma\_nv INT = SCOPE\_IDENTITY();

-- 3) Gán vai trò

IF (@VaiTro = N'TRUONG\_PHONG\_TC')

INSERT INTO dbo.TruongPhongTC(ma\_nv, quyen\_han) VALUES (@ma\_nv, N'Duyệt giao dịch');

ELSE IF (@VaiTro = N'NHAN\_VIEN\_TC')

INSERT INTO dbo.NhanVienTC(ma\_nv, chuyen\_mon, cap\_bac) VALUES (@ma\_nv, N'Tài chính', 'NhanVien');

ELSE IF (@VaiTro = N'KE\_TOAN')

INSERT INTO dbo.KeToan(ma\_nv, chung\_chi, kinh\_nghiem) VALUES (@ma\_nv, N'CPA', 0);

ELSE

BEGIN

RAISERROR(N'Vai trò không hợp lệ.', 16, 1);

ROLLBACK TRAN; RETURN;

END

-- 4) Provision SQL Login/User + gán DB role

IF (@Provision = 1)

EXEC dbo.SP\_ProvisionSqlAccount @Username, @Password;

COMMIT TRAN;

-- Trả về mã NV mới

SELECT @ma\_nv AS ma\_nv\_moi;

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int ThemNhanVien(

string hoTen,

string email,

string sdt,

string username,

string password,

string vaiTro,

bool provision = true)

{

var cmd = DbHelper.StoredProc("dbo.SP\_ThemNhanVien");

cmd.Parameters.Add(DbHelper.Param("@HoTen", hoTen));

cmd.Parameters.Add(DbHelper.Param("@Email", email));

cmd.Parameters.Add(DbHelper.Param("@Sdt", sdt));

cmd.Parameters.Add(DbHelper.Param("@Username", username));

cmd.Parameters.Add(DbHelper.Param("@Password", password));

cmd.Parameters.Add(DbHelper.Param("@VaiTro", vaiTro));

cmd.Parameters.Add(DbHelper.Param("@Provision", provision ? 1 : 0));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**e. Xóa nhân viên và vô hiệu hóa tài khoản của nhân viên đó**

CREATE OR ALTER PROCEDURE dbo.SP\_XoaNhanVien

@MaNv INT,

@HardDelete BIT = 0,

@DropSql BIT = 1

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRAN;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM dbo.NhanVien WHERE ma\_nv = @MaNv)

BEGIN

RAISERROR(N'Nhân viên không tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

DECLARE @username VARCHAR(50), @ma\_tk INT;

SELECT @ma\_tk = nv.ma\_tk, @username = tk.username

FROM dbo.NhanVien nv

JOIN dbo.TaiKhoan tk ON tk.ma\_tk = nv.ma\_tk

WHERE nv.ma\_nv = @MaNv;

IF (@HardDelete = 1)

BEGIN

-- Chặn nếu có tham chiếu ở GiaoDich

IF EXISTS (SELECT 1 FROM dbo.GiaoDich WHERE ma\_nv\_tao\_nvtc = @MaNv OR ma\_nv\_duyet\_tp = @MaNv)

BEGIN

RAISERROR(N'Không thể xóa vĩnh viễn do đang được tham chiếu trong giao dịch.', 16, 1);

ROLLBACK TRAN; RETURN;

END

-- Xóa NV trước (role tables ON DELETE CASCADE)

DELETE FROM dbo.NhanVien WHERE ma\_nv = @MaNv;

-- Xóa tài khoản ứng dụng

DELETE FROM dbo.TaiKhoan WHERE ma\_tk = @ma\_tk;

-- Drop SQL Login/User

IF (@DropSql = 1 AND @username IS NOT NULL)

EXEC dbo.SP\_DropSqlAccount @username;

END

ELSE

BEGIN

-- Soft delete: chỉ inactive tài khoản ứng dụng

UPDATE dbo.TaiKhoan SET trang\_thai = 'inactive' WHERE ma\_tk = @ma\_tk;

-- Drop SQL Login/User

IF (@DropSql = 1 AND @username IS NOT NULL)

EXEC dbo.SP\_DropSqlAccount @username;

END

COMMIT TRAN;

SELECT 1 AS success;

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int XoaNhanVien(int maNv, bool hardDelete = false, bool dropSql = true)

{

var cmd = DbHelper.StoredProc("dbo.SP\_XoaNhanVien");

cmd.Parameters.Add(DbHelper.Param("@MaNv", maNv));

cmd.Parameters.Add(DbHelper.Param("@HardDelete", hardDelete ? 1 : 0));

cmd.Parameters.Add(DbHelper.Param("@DropSql", dropSql ? 1 : 0));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**f. Kích hoạt lại tài khoản nhân viên bị vô hiệu hóa**

CREATE OR ALTER PROCEDURE dbo.SP\_KichHoatNhanVien

@MaNv INT,

@Provision BIT = 1

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRAN;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM dbo.NhanVien WHERE ma\_nv = @MaNv)

BEGIN

RAISERROR(N'Nhân viên không tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

DECLARE @ma\_tk INT, @username VARCHAR(50), @password VARCHAR(255);

SELECT @ma\_tk = nv.ma\_tk, @username = tk.username, @password = tk.[password]

FROM dbo.NhanVien nv

JOIN dbo.TaiKhoan tk ON tk.ma\_tk = nv.ma\_tk

WHERE nv.ma\_nv = @MaNv;

-- Kích hoạt lại tài khoản ứng dụng

UPDATE dbo.TaiKhoan SET trang\_thai = 'active' WHERE ma\_tk = @ma\_tk;

-- Cấp lại SQL Login/User + gán DB role theo FN\_Login\_GetRole

IF (@Provision = 1 AND @username IS NOT NULL AND @password IS NOT NULL)

EXEC dbo.SP\_ProvisionSqlAccount @username = @username, @password = @password;

COMMIT TRAN;

SELECT 1 AS success;

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int KichHoatNhanVien(int maNv, bool provision = true)

{

var cmd = DbHelper.StoredProc("dbo.SP\_KichHoatNhanVien");

cmd.Parameters.Add(DbHelper.Param("@MaNv", maNv));

cmd.Parameters.Add(DbHelper.Param("@Provision", provision ? 1 : 0));

var scalar = DbHelper.ExecuteScalar(cmd);

if (scalar == null || scalar == DBNull.Value) return 0;

return Convert.ToInt32(scalar);

}

**g. Thêm tài khoản ngân hàng**

CREATE PROCEDURE SP\_ThemTaiKhoanNH

@ten\_tk NVARCHAR(100),

@so\_tk VARCHAR(20),

@ngan\_hang NVARCHAR(100),

@so\_du DECIMAL(15,2) = 0

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra số tài khoản đã tồn tại chưa

IF EXISTS (SELECT 1 FROM TaiKhoanNH WHERE so\_tk = @so\_tk)

BEGIN

RAISERROR(N'Số tài khoản đã tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Thêm tài khoản mới

INSERT INTO TaiKhoanNH (ten\_tk, so\_tk, ngan\_hang, so\_du)

VALUES (@ten\_tk, @so\_tk, @ngan\_hang, @so\_du);

-- Trả về mã tài khoản mới

SELECT SCOPE\_IDENTITY() as ma\_tknh\_moi;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int ThemTaiKhoanNH(string tenTk, string soTk, string nganHang, decimal soDu = 0)

{

var cmd = DbHelper.StoredProc("dbo.SP\_ThemTaiKhoanNH");

cmd.Parameters.Add(DbHelper.Param("@ten\_tk", tenTk));

cmd.Parameters.Add(DbHelper.Param("@so\_tk", soTk));

cmd.Parameters.Add(DbHelper.Param("@ngan\_hang", nganHang));

cmd.Parameters.Add(DbHelper.Param("@so\_du", soDu));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**h. Sửa thông tin tài khoản ngân hàng**

CREATE PROCEDURE SP\_SuaTaiKhoanNH

@ma\_tknh INT,

@ten\_tk NVARCHAR(100),

@so\_tk VARCHAR(20),

@ngan\_hang NVARCHAR(100),

@trang\_thai VARCHAR(20) = 'active'

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra tài khoản có tồn tại không

IF NOT EXISTS (SELECT 1 FROM TaiKhoanNH WHERE ma\_tknh = @ma\_tknh)

BEGIN

RAISERROR(N'Tài khoản không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Kiểm tra số tài khoản mới có trùng không

IF EXISTS (SELECT 1 FROM TaiKhoanNH WHERE so\_tk = @so\_tk AND ma\_tknh != @ma\_tknh)

BEGIN

RAISERROR(N'Số tài khoản đã tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật tài khoản

UPDATE TaiKhoanNH

SET ten\_tk = @ten\_tk,

so\_tk = @so\_tk,

ngan\_hang = @ngan\_hang,

trang\_thai = @trang\_thai

WHERE ma\_tknh = @ma\_tknh;

-- Trả về số dòng được cập nhật

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int SuaTaiKhoanNH(int maTknh, string tenTk, string soTk, string nganHang, string trangThai = "active") {

var cmd = DbHelper.StoredProc("dbo.SP\_SuaTaiKhoanNH");

cmd.Parameters.Add(DbHelper.Param("@ma\_tknh", maTknh));

cmd.Parameters.Add(DbHelper.Param("@ten\_tk", tenTk));

cmd.Parameters.Add(DbHelper.Param("@so\_tk", soTk));

cmd.Parameters.Add(DbHelper.Param("@ngan\_hang", nganHang));

cmd.Parameters.Add(DbHelper.Param("@trang\_thai", trangThai));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**i. Vô hiệu hóa tài khoản ngân hàng**

CREATE PROCEDURE SP\_UpdateTaiKhoanNH

@ma\_tknh INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra tài khoản có tồn tại không

IF NOT EXISTS (SELECT 1 FROM TaiKhoanNH WHERE ma\_tknh = @ma\_tknh)

BEGIN

RAISERROR(N'Tài khoản không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật trạng thái tài khoản thành inactive

UPDATE TaiKhoanNH

SET trang\_thai = 'inactive'

WHERE ma\_tknh = @ma\_tknh;

-- Trả về số dòng được cập nhật

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int VohieuHoaTaiKhoanNH(int maTknh)

{

var cmd = DbHelper.StoredProc("dbo.SP\_UpdateTaiKhoanNH");

cmd.Parameters.Add(DbHelper.Param("@ma\_tknh", maTknh));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**j. Thêm dự án mới**

CREATE PROCEDURE SP\_ThemDuAn

@ten\_du\_an NVARCHAR(200),

@ngay\_bd DATE,

@ngay\_kt DATE = NULL,

@ngan\_sach DECIMAL(15,2) = 0

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Thêm dự án mới

INSERT INTO DuAn (ten\_du\_an, ngay\_bd, ngay\_kt, ngan\_sach)

VALUES (@ten\_du\_an, @ngay\_bd, @ngay\_kt, @ngan\_sach);

-- Trả về mã dự án mới

SELECT SCOPE\_IDENTITY() as ma\_du\_an\_moi;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int ThemDuAn(string tenDuAn, DateTime ngayBd, DateTime? ngayKt, decimal nganSach = 0)

{

var cmd = DbHelper.StoredProc("dbo.SP\_ThemDuAn");

cmd.Parameters.Add(DbHelper.Param("@ten\_du\_an", tenDuAn));

cmd.Parameters.Add(DbHelper.Param("@ngay\_bd", ngayBd));

cmd.Parameters.Add(DbHelper.Param("@ngay\_kt", ngayKt));

cmd.Parameters.Add(DbHelper.Param("@ngan\_sach", nganSach));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**k. Sửa thông tin dự án**

CREATE PROCEDURE SP\_SuaDuAn

@ma\_du\_an INT,

@ten\_du\_an NVARCHAR(200),

@ngay\_bd DATE,

@ngay\_kt DATE = NULL,

@ngan\_sach DECIMAL(15,2),

@trang\_thai VARCHAR(20) = 'active'

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra dự án có tồn tại không

IF NOT EXISTS (SELECT 1 FROM DuAn WHERE ma\_du\_an = @ma\_du\_an)

BEGIN

RAISERROR(N'Dự án không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật dự án

UPDATE DuAn

SET ten\_du\_an = @ten\_du\_an,

ngay\_bd = @ngay\_bd,

ngay\_kt = @ngay\_kt,

ngan\_sach = @ngan\_sach,

trang\_thai = @trang\_thai

WHERE ma\_du\_an = @ma\_du\_an;

-- Trả về số dòng được cập nhật

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int SuaDuAn(int maDuAn, string tenDuAn, DateTime ngayBd, DateTime? ngayKt, decimal nganSach, string trangThai = "active")

{

var cmd = DbHelper.StoredProc("dbo.SP\_SuaDuAn");

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

cmd.Parameters.Add(DbHelper.Param("@ten\_du\_an", tenDuAn));

cmd.Parameters.Add(DbHelper.Param("@ngay\_bd", ngayBd));

cmd.Parameters.Add(DbHelper.Param("@ngay\_kt", ngayKt));

cmd.Parameters.Add(DbHelper.Param("@ngan\_sach", nganSach));

cmd.Parameters.Add(DbHelper.Param("@trang\_thai", trangThai));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

**l. Vô hiệu hóa dự án**

CREATE PROCEDURE SP\_UpdateDuAn

@ma\_du\_an INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

-- Kiểm tra dự án có tồn tại không

IF NOT EXISTS (SELECT 1 FROM DuAn WHERE ma\_du\_an = @ma\_du\_an)

BEGIN

RAISERROR(N'Dự án không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

-- Cập nhật trạng thái dự án thành inactive

UPDATE DuAn

SET trang\_thai = 'inactive'

WHERE ma\_du\_an = @ma\_du\_an;

-- Trả về số dòng được cập nhật

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**Xử lý bên C#:**

public int VohieuHoaDuAn(int maDuAn)

{

var cmd = DbHelper.StoredProc("dbo.SP\_UpdateDuAn");

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

var scalar = DbHelper.ExecuteScalar(cmd);

return Convert.ToInt32(scalar);

}

## **4. Chức năng chung của trưởng phòng tài chính và kế toán**

**a. Tra cứu toàn bộ lịch sử giao dịch**

CREATE OR ALTER PROCEDURE dbo.SP\_GetLichSuGiaoDich

@TuNgay DATETIME = NULL,

@DenNgay DATETIME = NULL,

@TrangThai VARCHAR(20) = NULL,

@LoaiGd VARCHAR(10) = NULL,

@MaDuAn INT = NULL,

@MaTknh INT = NULL,

@MaNvTao INT = NULL

AS

BEGIN

SET NOCOUNT ON;

SELECT

v.ma\_gd,

v.loai\_gd,

v.so\_tien,

v.ngay\_gd,

v.mo\_ta,

v.trang\_thai,

v.ten\_loai,

v.tai\_khoan,

v.nguoi\_tao,

v.nguoi\_duyet,

v.ngay\_duyet,

v.ten\_du\_an

FROM dbo.V\_LichSuGiaoDich AS v

INNER JOIN dbo.GiaoDich AS gd

ON gd.ma\_gd = v.ma\_gd

WHERE (@TuNgay IS NULL OR v.ngay\_gd >= @TuNgay)

AND (@DenNgay IS NULL OR v.ngay\_gd < DATEADD(DAY, 1, @DenNgay))

AND (@TrangThai IS NULL OR v.trang\_thai = @TrangThai)

AND (@LoaiGd IS NULL OR v.loai\_gd = @LoaiGd)

AND (@MaDuAn IS NULL OR gd.ma\_du\_an = @MaDuAn)

AND (@MaTknh IS NULL OR gd.ma\_tknh = @MaTknh)

AND (@MaNvTao IS NULL OR gd.ma\_nv\_tao\_nvtc = @MaNvTao)

ORDER BY v.ngay\_gd DESC;

END

**Xử lý bên C#:**

public DataTable GetLichSuGiaoDich(DateTime? tuNgay = null, DateTime? denNgay = null, string trangThai = null, string loaiGd = null, int? maDuAn = null, int? maTknh = null, int? maNvTao = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_GetLichSuGiaoDich");

cmd.Parameters.Add(DbHelper.Param("@TuNgay", tuNgay));

cmd.Parameters.Add(DbHelper.Param("@DenNgay", denNgay));

cmd.Parameters.Add(DbHelper.Param("@TrangThai", trangThai));

cmd.Parameters.Add(DbHelper.Param("@LoaiGd", loaiGd));

cmd.Parameters.Add(DbHelper.Param("@MaDuAn", maDuAn));

cmd.Parameters.Add(DbHelper.Param("@MaTknh", maTknh));

cmd.Parameters.Add(DbHelper.Param("@MaNvTao", maNvTao));

return DbHelper.ExecuteDataTable(cmd);

}

**b. Thống kê tình hình tài chính theo tháng trong năm**

CREATE PROCEDURE SP\_GetThongKeThang

@Nam INT = NULL,

@Thang INT = NULL

AS

BEGIN

SET NOCOUNT ON;

SELECT

Nam,

Thang,

TongThu,

TongChi,

LaiLo

FROM V\_ThongKeThang

WHERE (@Nam IS NULL OR Nam = @Nam)

AND (@Thang IS NULL OR Thang = @Thang)

ORDER BY Nam DESC, Thang DESC;

END;

**Xử lý bên C#:**

public DataTable GetThongKeThang(int? nam = null, int? thang = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_GetThongKeThang");

cmd.Parameters.Add(DbHelper.Param("@Nam", nam));

cmd.Parameters.Add(DbHelper.Param("@Thang", thang));

return DbHelper.ExecuteDataTable(cmd);

}

**c. Xuất báo cáo chi tiết về các giao dịch thu chi phát sinh của dự án trong một khoảng thời gian**

CREATE PROCEDURE SP\_XuatBaoCaoChiTiet

@ngay\_bd DATETIME,

@ngay\_kt DATETIME,

@ma\_du\_an INT = NULL

AS

BEGIN

SET NOCOUNT ON;

SELECT

gd.ma\_gd,

gd.loai\_gd,

gd.so\_tien,

gd.ngay\_gd,

gd.mo\_ta,

gd.trang\_thai,

lg.ten\_loai,

tkn.ten\_tk as tai\_khoan,

nv.ho\_ten as nguoi\_tao,

nv\_duyet.ho\_ten as nguoi\_duyet,

gd.ngay\_duyet,

da.ten\_du\_an

FROM GiaoDich gd

INNER JOIN LoaiGiaoDich lg ON gd.ma\_loai = lg.ma\_loai

INNER JOIN TaiKhoanNH tkn ON gd.ma\_tknh = tkn.ma\_tknh

INNER JOIN NhanVienTC nvtc ON gd.ma\_nv\_tao\_nvtc = nvtc.ma\_nv

INNER JOIN NhanVien nv ON nvtc.ma\_nv = nv.ma\_nv

LEFT JOIN TruongPhongTC tp\_duyet ON tp\_duyet.ma\_nv = gd.ma\_nv\_duyet\_tp

LEFT JOIN NhanVien nv\_duyet ON nv\_duyet.ma\_nv = tp\_duyet.ma\_nv

LEFT JOIN DuAn da ON gd.ma\_du\_an = da.ma\_du\_an

WHERE gd.ngay\_gd >= @ngay\_bd

AND gd.ngay\_gd < DATEADD(DAY, 1, @ngay\_kt)

AND (@ma\_du\_an IS NULL OR gd.ma\_du\_an = @ma\_du\_an)

ORDER BY gd.ngay\_gd DESC;

END;

**Xử lý bên C#:**

public DataTable XuatBaoCaoChiTiet(DateTime ngayBd, DateTime ngayKt, int? maDuAn = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_XuatBaoCaoChiTiet");

cmd.Parameters.Add(DbHelper.Param("@ngay\_bd", ngayBd));

cmd.Parameters.Add(DbHelper.Param("@ngay\_kt", ngayKt));

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

return DbHelper.ExecuteDataTable(cmd);

}

**d. Tính tổng thu và tổng chi trong một khoảng thời gian**

CREATE FUNCTION FN\_TinhTongThuChi(

@ngay\_bd DATETIME,

@ngay\_kt DATETIME,

@ma\_du\_an INT = NULL

)

RETURNS TABLE

AS

RETURN

(

SELECT

loai\_gd,

SUM(so\_tien) as tong\_tien,

COUNT(\*) as so\_giao\_dich

FROM GiaoDich

WHERE ngay\_gd >= @ngay\_bd

AND ngay\_gd < DATEADD(DAY, 1, @ngay\_kt)

AND trang\_thai = 'DA\_DUYET'

AND (@ma\_du\_an IS NULL OR ma\_du\_an = @ma\_du\_an)

GROUP BY loai\_gd

);

**Xử lý bên C#:**

public DataTable TinhTongThuChi(DateTime ngayBd, DateTime ngayKt, int? maDuAn = null)

{

using (var cmd = new SqlCommand(

"SELECT \* FROM dbo.FN\_TinhTongThuChi(@ngay\_bd, @ngay\_kt, @ma\_du\_an);")

{ CommandType = CommandType.Text })

{

cmd.Parameters.Add(DbHelper.Param("@ngay\_bd", ngayBd));

cmd.Parameters.Add(DbHelper.Param("@ngay\_kt", ngayKt));

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

return DbHelper.ExecuteDataTable(cmd);

}

}

**e. Tính lãi lỗ**

CREATE FUNCTION FN\_TinhLaiLo(

@ngay\_bd DATETIME,

@ngay\_kt DATETIME,

@ma\_du\_an INT = NULL

)

RETURNS DECIMAL(15,2)

AS

BEGIN

DECLARE @tong\_thu DECIMAL(15,2) = 0;

DECLARE @tong\_chi DECIMAL(15,2) = 0;

-- Tính tổng thu chi từ giao dịch đã duyệt

SELECT

@tong\_thu = ISNULL(SUM(CASE WHEN loai\_gd = 'THU' THEN so\_tien ELSE 0 END), 0),

@tong\_chi = ISNULL(SUM(CASE WHEN loai\_gd = 'CHI' THEN so\_tien ELSE 0 END), 0)

FROM GiaoDich

WHERE ngay\_gd >= @ngay\_bd

AND ngay\_gd < DATEADD(DAY, 1, @ngay\_kt)

AND trang\_thai = 'DA\_DUYET'

AND (@ma\_du\_an IS NULL OR ma\_du\_an = @ma\_du\_an);

-- Trả về lãi/lỗ (thu - chi)

RETURN @tong\_thu - @tong\_chi;

END;

**Xử lý bên C#:**

public decimal TinhLaiLo(DateTime ngayBd, DateTime ngayKt, int? maDuAn = null)

{

using (var cmd = new SqlCommand(

"SELECT dbo.FN\_TinhLaiLo(@ngay\_bd, @ngay\_kt, @ma\_du\_an);")

{ CommandType = CommandType.Text })

{

cmd.Parameters.Add(DbHelper.Param("@ngay\_bd", ngayBd));

cmd.Parameters.Add(DbHelper.Param("@ngay\_kt", ngayKt));

cmd.Parameters.Add(DbHelper.Param("@ma\_du\_an", maDuAn));

var scalar = DbHelper.ExecuteScalar(cmd);

return scalar == null || scalar == DBNull.Value ? 0m : Convert.ToDecimal(scalar);

}

}

## **5. Chức năng chung của nhân viên tài chính, trưởng phòng và kế toán**

**a. Tra cứu thông tin dự án**

CREATE PROCEDURE SP\_GetDuAn

@TrangThai VARCHAR(20) = NULL,

@TuNgayBd DATETIME = NULL,

@DenNgayBd DATETIME = NULL,

@CoNganSach BIT = NULL

AS BEGIN

SET NOCOUNT ON;

SELECT

ma\_du\_an,

ten\_du\_an,

ngay\_bd,

ngay\_kt,

ngan\_sach,

trang\_thai

FROM V\_DuAn

WHERE (@TrangThai IS NULL OR trang\_thai = @TrangThai)

AND (@TuNgayBd IS NULL OR ngay\_bd >= @TuNgayBd)

AND (@DenNgayBd IS NULL OR ngay\_bd <= @DenNgayBd)

AND (@CoNganSach IS NULL OR

CASE WHEN ngan\_sach > 0 THEN CAST(1 AS BIT) ELSE CAST(0 AS BIT) END = @CoNganSach)

ORDER BY ngay\_bd DESC;

END;

**Xử lý bên C#:**

public DataTable GetDuAn(string trangThai = null, DateTime? tuNgayBd = null, DateTime? denNgayBd = null, bool? coNganSach = null)

{

var cmd = DbHelper.StoredProc("dbo.SP\_GetDuAn");

cmd.Parameters.Add(DbHelper.Param("@TrangThai", trangThai));

cmd.Parameters.Add(DbHelper.Param("@TuNgayBd", tuNgayBd));

cmd.Parameters.Add(DbHelper.Param("@DenNgayBd", denNgayBd));

cmd.Parameters.Add(DbHelper.Param("@CoNganSach", coNganSach.HasValue ? (object)(coNganSach.Value ? 1 : 0) : DBNull.Value));

return DbHelper.ExecuteDataTable(cmd);

}

**b. Tra cứu thông tin tài khoản ngân hàng**

CREATE PROCEDURE SP\_GetTaiKhoanNH

@TrangThai VARCHAR(20) = NULL,

@NganHang NVARCHAR(100) = NULL,

@CoSoDu BIT = NULL

AS BEGIN

SET NOCOUNT ON;

SELECT

ma\_tknh,

ten\_tk,

so\_tk,

ngan\_hang,

so\_du,

trang\_thai

FROM V\_TaiKhoanNH

WHERE (@TrangThai IS NULL OR trang\_thai = @TrangThai)

AND (@NganHang IS NULL OR ngan\_hang LIKE '%' + @NganHang + '%')

AND (@CoSoDu IS NULL OR

CASE WHEN so\_du > 0 THEN CAST(1 AS BIT) ELSE CAST(0 AS BIT) END = @CoSoDu)

ORDER BY so\_du DESC;

END;

**Xử lý bên C#:**

public DataTable GetTaiKhoanNH(string trangThai = null, string nganHang = null, bool? coSoDu = null) {

var cmd = DbHelper.StoredProc("dbo.SP\_GetTaiKhoanNH");

cmd.Parameters.Add(DbHelper.Param("@TrangThai", trangThai));

cmd.Parameters.Add(DbHelper.Param("@NganHang", nganHang));

cmd.Parameters.Add(DbHelper.Param("@CoSoDu", coSoDu.HasValue ? (object)(coSoDu.Value ? 1 : 0) : DBNull.Value));

return DbHelper.ExecuteDataTable(cmd);

}

**c. Tra cứu thông tin loại giao dịch**

CREATE PROCEDURE SP\_GetLoaiGiaoDich @LoaiThuChi VARCHAR(10) = NULL

AS BEGIN

SET NOCOUNT ON;

SELECT

ma\_loai,

ten\_loai,

loai\_thu\_chi,

mo\_ta

FROM V\_LoaiGiaoDich

WHERE (@LoaiThuChi IS NULL OR loai\_thu\_chi = @LoaiThuChi)

ORDER BY loai\_thu\_chi, ten\_loai;

END;

**Xử lý bên C#:**

public DataTable GetLoaiGiaoDich(string loaiThuChi = null) {

var cmd = DbHelper.StoredProc("dbo.SP\_GetLoaiGiaoDich");

cmd.Parameters.Add(DbHelper.Param("@LoaiThuChi", loaiThuChi));

return DbHelper.ExecuteDataTable(cmd);

}

# **CHƯƠNG 4: PHÂN QUYỀN**

## **1. Tạo role**

Trong hệ thống quản lý thu chi – tài chính của trung tâm giới thiệu việc làm sinh viên có ba role chính đó là trưởng phòng tài chính, nhân viên tài chính và kế toán. Tương ứng sẽ có ba role trong SQL Server và mỗi role sẽ có một quyền riêng và chung thông qua sự phân quyền

CREATE ROLE rl\_truongphong AUTHORIZATION dbo;

CREATE ROLE rl\_nhanvien\_tc AUTHORIZATION dbo;

CREATE ROLE rl\_ketoan AUTHORIZATION dbo;

## **2. Phân quyền**

**Gán quyền cho role rl\_truongphong**

GRANT SELECT ON LoaiGiaoDich TO rl\_truongphong;

GRANT SELECT, INSERT, UPDATE ON TaiKhoanNH TO rl\_truongphong;

GRANT SELECT, INSERT, UPDATE ON DuAn TO rl\_truongphong;

GRANT SELECT ON GiaoDich TO rl\_truongphong;

GRANT EXECUTE ON SP\_GetThongKeThang TO rl\_truongphong;

GRANT EXECUTE ON SP\_GetGiaoDichChoDuyet TO rl\_truongphong;

GRANT EXECUTE ON SP\_GetLichSuGiaoDich TO rl\_truongphong;

GRANT EXECUTE ON SP\_GetDuAn TO rl\_truongphong;

GRANT EXECUTE ON SP\_GetTaiKhoanNH TO rl\_truongphong;

GRANT EXECUTE ON SP\_GetLoaiGiaoDich TO rl\_truongphong;

GRANT EXECUTE ON SP\_DuyetGiaoDich TO rl\_truongphong;

GRANT EXECUTE ON SP\_XuatBaoCaoChiTiet TO rl\_truongphong;

GRANT EXECUTE ON SP\_ThemDuAn TO rl\_truongphong;

GRANT EXECUTE ON SP\_SuaDuAn TO rl\_truongphong;

GRANT EXECUTE ON SP\_UpdateDuAn TO rl\_truongphong;

GRANT EXECUTE ON SP\_ThemTaiKhoanNH TO rl\_truongphong;

GRANT EXECUTE ON SP\_SuaTaiKhoanNH TO rl\_truongphong;

GRANT EXECUTE ON SP\_UpdateTaiKhoanNH TO rl\_truongphong;

GRANT EXECUTE ON SP\_ProvisionSqlAccount TO rl\_truongphong;

GRANT EXECUTE ON SP\_DropSqlAccount TO rl\_truongphong;

GRANT EXECUTE ON dbo.SP\_KichHoatNhanVien TO rl\_truongphong;

GRANT EXECUTE ON dbo.SP\_TimNhanVien TO rl\_truongphong;

GRANT EXECUTE ON dbo.SP\_ThemNhanVien TO rl\_truongphong;

GRANT EXECUTE ON dbo.SP\_XoaNhanVien TO rl\_truongphong;

GRANT EXECUTE ON FN\_TinhLaiLo TO rl\_truongphong;

GRANT EXECUTE ON FN\_KiemTraSoDu TO rl\_truongphong;

GRANT SELECT ON FN\_TinhTongThuChi TO rl\_truongphong;

GRANT SELECT ON dbo.FN\_Login\_GetRole TO rl\_truongphong;

**Gán quyền cho role rl\_nhanvien\_tc**

GRANT SELECT ON LoaiGiaoDich TO rl\_nhanvien\_tc;

GRANT SELECT ON TaiKhoanNH TO rl\_nhanvien\_tc;

GRANT SELECT ON DuAn TO rl\_nhanvien\_tc;

GRANT SELECT, INSERT ON GiaoDich TO rl\_nhanvien\_tc;

GRANT EXECUTE ON SP\_GetLichSuGiaoDich\_ForNhanVienTC TO rl\_nhanvien\_tc;

GRANT EXECUTE ON SP\_GetDuAn TO rl\_nhanvien\_tc;

GRANT EXECUTE ON SP\_GetTaiKhoanNH TO rl\_nhanvien\_tc;

GRANT EXECUTE ON SP\_GetLoaiGiaoDich TO rl\_nhanvien\_tc;

GRANT EXECUTE ON SP\_ThemGiaoDich TO rl\_nhanvien\_tc;

GRANT EXECUTE ON SP\_SuaGiaoDich TO rl\_nhanvien\_tc;

GRANT EXECUTE ON dbo.SP\_TimNhanVien TO rl\_nhanvien\_tc;

GRANT EXECUTE ON FN\_KiemTraSoDu TO rl\_nhanvien\_tc;

GRANT SELECT ON dbo.FN\_Login\_GetRole TO rl\_nhanvien\_tc;

**Gán quyền cho role rl\_ketoan**

GRANT SELECT ON LoaiGiaoDich TO rl\_ketoan;

GRANT SELECT ON TaiKhoanNH TO rl\_ketoan;

GRANT SELECT ON DuAn TO rl\_ketoan;

GRANT SELECT ON GiaoDich TO rl\_ketoan;

GRANT EXECUTE ON SP\_GetDuAn TO rl\_ketoan;

GRANT EXECUTE ON SP\_GetTaiKhoanNH TO rl\_ketoan;

GRANT EXECUTE ON SP\_GetLoaiGiaoDich TO rl\_ketoan;

GRANT EXECUTE ON SP\_GetThongKeThang TO rl\_ketoan;

GRANT EXECUTE ON SP\_GetLichSuGiaoDich TO rl\_ketoan;

GRANT EXECUTE ON SP\_XuatBaoCaoChiTiet TO rl\_ketoan;

GRANT EXECUTE ON FN\_TinhLaiLo TO rl\_ketoan;

GRANT EXECUTE ON FN\_KiemTraSoDu TO rl\_ketoan;

GRANT SELECT ON FN\_TinhTongThuChi TO rl\_ketoan;

GRANT SELECT ON dbo.FN\_Login\_GetRole TO rl\_ketoan;

GRANT EXECUTE ON dbo.SP\_TimNhanVien TO rl\_ketoan;

**Procedure tạo tài khoản SQL**

CREATE OR ALTER PROCEDURE SP\_ProvisionSqlAccount

@username VARCHAR(50),

@password VARCHAR(255)

WITH EXECUTE AS OWNER

AS

BEGIN

SET NOCOUNT ON;

DECLARE @sql NVARCHAR(MAX), @role VARCHAR(50);

-- Tạo LOGIN nếu chưa có (yêu cầu securityadmin/sysadmin)

IF NOT EXISTS (SELECT 1 FROM sys.server\_principals WHERE name = @username)

BEGIN

SET @sql = N'CREATE LOGIN [' + @username + N'] WITH PASSWORD = ' + QUOTENAME(@password,'''') + N', CHECK\_POLICY = OFF, CHECK\_EXPIRATION = OFF;';

EXEC (@sql);

END

-- Tạo USER nếu chưa có trong database hiện tại

IF NOT EXISTS (SELECT 1 FROM sys.database\_principals WHERE name = @username)

BEGIN

SET @sql = N'CREATE USER [' + @username + N'] FOR LOGIN [' + @username + N'];';

EXEC (@sql);

END

-- Gán ROLE dựa vào hàm xác định vai trò

SELECT TOP 1 @role = vai\_tro FROM dbo.FN\_Login\_GetRole(@username, @password);

IF (@role IS NULL OR @role = 'UNKNOWN')

BEGIN

RAISERROR(N'Không tìm thấy vai trò ứng với tài khoản trong dữ liệu ứng dụng.', 16, 1);

RETURN;

END

IF @role = 'TRUONG\_PHONG\_TC' EXEC(N'ALTER ROLE rl\_truongphong ADD MEMBER [' + @username + N']');

ELSE IF @role = 'NHAN\_VIEN\_TC' EXEC(N'ALTER ROLE rl\_nhanvien\_tc ADD MEMBER [' + @username + N']');

ELSE IF @role = 'KE\_TOAN' EXEC(N'ALTER ROLE rl\_ketoan ADD MEMBER [' + @username + N']');

END;

**Procedure xóa tài khoản SQL**

CREATE OR ALTER PROCEDURE SP\_DropSqlAccount

@username VARCHAR(50)

WITH EXECUTE AS OWNER

AS

BEGIN

SET NOCOUNT ON;

DECLARE @sql NVARCHAR(MAX);

-- Xóa USER trong DB (nếu tồn tại)

IF EXISTS (SELECT 1 FROM sys.database\_principals WHERE name = @username)

BEGIN TRY

SET @sql = N'DROP USER [' + @username + N']';

EXEC (@sql);

END TRY BEGIN CATCH END CATCH;

-- Xóa LOGIN ở server (nếu tồn tại)

IF EXISTS (SELECT 1 FROM sys.server\_principals WHERE name = @username)

BEGIN TRY

SET @sql = N'DROP LOGIN [' + @username + N']';

EXEC (@sql);

END TRY BEGIN CATCH END CATCH;

END;

**Function lấy role cho việc đăng nhập vào hệ thống**

CREATE OR ALTER FUNCTION dbo.FN\_Login\_GetRole

(

@username NVARCHAR(50),

@password NVARCHAR(255)

)

RETURNS TABLE

AS

RETURN

(

WITH u AS (

SELECT nv.ma\_nv, tk.username

FROM dbo.TaiKhoan tk

JOIN dbo.NhanVien nv ON nv.ma\_tk = tk.ma\_tk

WHERE tk.username = @username

AND tk.[password] = @password

AND tk.trang\_thai = 'active'

)

SELECT TOP (1)

CASE

WHEN EXISTS (

SELECT 1

FROM sys.database\_role\_members drm

JOIN sys.database\_principals r ON r.principal\_id = drm.role\_principal\_id

JOIN sys.database\_principals m ON m.principal\_id = drm.member\_principal\_id

WHERE r.[name] = N'rl\_truongphong' AND m.[name] = @username

) THEN N'TRUONG\_PHONG\_TC'

WHEN EXISTS (

SELECT 1

FROM sys.database\_role\_members drm

JOIN sys.database\_principals r ON r.principal\_id = drm.role\_principal\_id

JOIN sys.database\_principals m ON m.principal\_id = drm.member\_principal\_id

WHERE r.[name] = N'rl\_nhanvien\_tc' AND m.[name] = @username

) THEN N'NHAN\_VIEN\_TC'

WHEN EXISTS (

SELECT 1

FROM sys.database\_role\_members drm

JOIN sys.database\_principals r ON r.principal\_id = drm.role\_principal\_id

JOIN sys.database\_principals m ON m.principal\_id = drm.member\_principal\_id

WHERE r.[name] = N'rl\_ketoan' AND m.[name] = @username

) THEN N'KE\_TOAN'

WHEN EXISTS (SELECT 1 FROM dbo.TruongPhongTC tp WHERE tp.ma\_nv = u.ma\_nv) THEN N'TRUONG\_PHONG\_TC'

WHEN EXISTS (SELECT 1 FROM dbo.NhanVienTC nv2 WHERE nv2.ma\_nv = u.ma\_nv) THEN N'NHAN\_VIEN\_TC'

WHEN EXISTS (SELECT 1 FROM dbo.KeToan kt WHERE kt.ma\_nv = u.ma\_nv) THEN N'KE\_TOAN'

ELSE N'UNKNOWN'

END AS vai\_tro,

u.ma\_nv

FROM u

);

**Xử lý bên C#:**

public async Task<(string role, int maNv)> GetRoleAsync(string username, string password)

{

try

{

using (var conn = DatabaseConnection.CreateConnection())

using (var cmd = new SqlCommand(

"SELECT vai\_tro, ma\_nv FROM dbo.FN\_Login\_GetRole(@u,@p)", conn))

{

cmd.Parameters.AddWithValue("@u", username);

cmd.Parameters.AddWithValue("@p", password);

await conn.OpenAsync();

using (var rd = await cmd.ExecuteReaderAsync(CommandBehavior.SingleRow))

{

if (await rd.ReadAsync())

return (rd.GetString(0), rd.GetInt32(1));

return (null, 0);

}

}

}

catch (Exception ex)

{

throw new DataAccessException("Lỗi truy vấn FN\_Login\_GetRole", ex);

}

}

# **CHƯƠNG 5: TRANSACTION**

**a. Transaction duyệt giao dịch của trưởng phòng**

CREATE PROCEDURE SP\_DuyetGiaoDich

@ma\_gd INT,

@ma\_nv\_duyet\_tp INT,

@trang\_thai VARCHAR(20)

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM GiaoDich WHERE ma\_gd = @ma\_gd AND trang\_thai = 'CHO\_DUYET')

BEGIN

RAISERROR(N'Giao dịch không tồn tại hoặc không ở trạng thái chờ duyệt!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

IF NOT EXISTS (

SELECT 1 FROM TruongPhongTC tp

WHERE tp.ma\_nv = @ma\_nv\_duyet\_tp

)

BEGIN

RAISERROR(N'Không có quyền duyệt giao dịch!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

DECLARE @loai\_gd\_cur VARCHAR(10);

DECLARE @ma\_tknh\_cur INT;

DECLARE @so\_tien\_cur DECIMAL(15,2);

SELECT @loai\_gd\_cur = loai\_gd, @ma\_tknh\_cur = ma\_tknh, @so\_tien\_cur = so\_tien

FROM GiaoDich WHERE ma\_gd = @ma\_gd;

IF @trang\_thai = 'DA\_DUYET' AND @loai\_gd\_cur = 'CHI'

BEGIN

DECLARE @so\_du\_hien\_tai DECIMAL(15,2);

SELECT @so\_du\_hien\_tai = so\_du FROM TaiKhoanNH WHERE ma\_tknh = @ma\_tknh\_cur;

IF @so\_du\_hien\_tai < @so\_tien\_cur

BEGIN

RAISERROR(N'Số dư tài khoản không đủ để duyệt giao dịch chi!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

END

UPDATE GiaoDich

SET trang\_thai = @trang\_thai,

ma\_nv\_duyet\_tp = @ma\_nv\_duyet\_tp,

ngay\_duyet = GETDATE()

WHERE ma\_gd = @ma\_gd;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**b. Transaction thêm giao dịch của nhân viên tài chính**

CREATE PROCEDURE SP\_ThemGiaoDich

@loai\_gd VARCHAR(10),

@so\_tien DECIMAL(15,2),

@mo\_ta NVARCHAR(500),

@ma\_loai VARCHAR(10),

@ma\_tknh INT,

@ma\_nv\_tao\_nvtc INT,

@ma\_du\_an INT = NULL

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (

SELECT 1

FROM LoaiGiaoDich lgd

WHERE lgd.ma\_loai = @ma\_loai

AND lgd.loai\_thu\_chi = @loai\_gd

)

BEGIN

RAISERROR(N'Loại giao dịch không khớp THU/CHI với mã loại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

INSERT INTO GiaoDich (loai\_gd, so\_tien, mo\_ta, ma\_loai, ma\_tknh, ma\_nv\_tao\_nvtc, ma\_du\_an)

VALUES (@loai\_gd, @so\_tien, @mo\_ta, @ma\_loai, @ma\_tknh, @ma\_nv\_tao\_nvtc, @ma\_du\_an);

SELECT SCOPE\_IDENTITY() as ma\_gd\_moi;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**c. Transaction sửa giao dịch chờ duyệt của nhân viên tài chính**

CREATE PROCEDURE SP\_SuaGiaoDich

@ma\_gd INT,

@so\_tien DECIMAL(15,2),

@mo\_ta NVARCHAR(500),

@ma\_loai VARCHAR(10),

@ma\_tknh INT,

@ma\_du\_an INT = NULL,

@ma\_nv\_sua INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM GiaoDich WHERE ma\_gd = @ma\_gd AND trang\_thai = 'CHO\_DUYET')

BEGIN

RAISERROR(N'Giao dịch không tồn tại hoặc đã được duyệt!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

IF NOT EXISTS (SELECT 1 FROM GiaoDich WHERE ma\_gd = @ma\_gd AND ma\_nv\_tao\_nvtc = @ma\_nv\_sua)

BEGIN

RAISERROR(N'Không có quyền sửa giao dịch của người khác!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

DECLARE @loai\_gd\_hien\_tai VARCHAR(10);

SELECT @loai\_gd\_hien\_tai = loai\_gd FROM GiaoDich WHERE ma\_gd = @ma\_gd;

IF NOT EXISTS (

SELECT 1

FROM LoaiGiaoDich lgd

WHERE lgd.ma\_loai = @ma\_loai

AND lgd.loai\_thu\_chi = @loai\_gd\_hien\_tai

)

BEGIN

RAISERROR(N'Loại giao dịch không khớp THU/CHI với mã loại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE GiaoDich

SET so\_tien = @so\_tien,

mo\_ta = @mo\_ta,

ma\_loai = @ma\_loai,

ma\_tknh = @ma\_tknh,

ma\_du\_an = @ma\_du\_an

WHERE ma\_gd = @ma\_gd;

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**d. Transaction thêm dự án**

CREATE PROCEDURE SP\_ThemDuAn

@ten\_du\_an NVARCHAR(200),

@ngay\_bd DATE,

@ngay\_kt DATE = NULL,

@ngan\_sach DECIMAL(15,2) = 0

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

INSERT INTO DuAn (ten\_du\_an, ngay\_bd, ngay\_kt, ngan\_sach)

VALUES (@ten\_du\_an, @ngay\_bd, @ngay\_kt, @ngan\_sach);

SELECT SCOPE\_IDENTITY() as ma\_du\_an\_moi;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**e. Transaction sửa dự án**

CREATE PROCEDURE SP\_SuaDuAn

@ma\_du\_an INT,

@ten\_du\_an NVARCHAR(200),

@ngay\_bd DATE,

@ngay\_kt DATE = NULL,

@ngan\_sach DECIMAL(15,2),

@trang\_thai VARCHAR(20) = 'active'

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM DuAn WHERE ma\_du\_an = @ma\_du\_an)

BEGIN

RAISERROR(N'Dự án không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE DuAn

SET ten\_du\_an = @ten\_du\_an,

ngay\_bd = @ngay\_bd,

ngay\_kt = @ngay\_kt,

ngan\_sach = @ngan\_sach,

trang\_thai = @trang\_thai

WHERE ma\_du\_an = @ma\_du\_an;

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**f. Transaction vô hiệu hóa dự án**

CREATE PROCEDURE SP\_UpdateDuAn

@ma\_du\_an INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM DuAn WHERE ma\_du\_an = @ma\_du\_an)

BEGIN

RAISERROR(N'Dự án không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE DuAn

SET trang\_thai = 'inactive'

WHERE ma\_du\_an = @ma\_du\_an;

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**g. Transaction thêm tài khoản ngân hàng**

CREATE PROCEDURE SP\_ThemTaiKhoanNH

@ten\_tk NVARCHAR(100),

@so\_tk VARCHAR(20),

@ngan\_hang NVARCHAR(100),

@so\_du DECIMAL(15,2) = 0

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF EXISTS (SELECT 1 FROM TaiKhoanNH WHERE so\_tk = @so\_tk)

BEGIN

RAISERROR(N'Số tài khoản đã tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

INSERT INTO TaiKhoanNH (ten\_tk, so\_tk, ngan\_hang, so\_du)

VALUES (@ten\_tk, @so\_tk, @ngan\_hang, @so\_du);

SELECT SCOPE\_IDENTITY() as ma\_tknh\_moi;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**h. Transaction sửa tài khoản ngân hàng**

CREATE PROCEDURE SP\_SuaTaiKhoanNH

@ma\_tknh INT,

@ten\_tk NVARCHAR(100),

@so\_tk VARCHAR(20),

@ngan\_hang NVARCHAR(100),

@trang\_thai VARCHAR(20) = 'active'

AS BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM TaiKhoanNH WHERE ma\_tknh = @ma\_tknh) BEGIN

RAISERROR(N'Tài khoản không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

IF EXISTS (SELECT 1 FROM TaiKhoanNH WHERE so\_tk = @so\_tk AND ma\_tknh != @ma\_tknh) BEGIN

RAISERROR(N'Số tài khoản đã tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE TaiKhoanNH

SET ten\_tk = @ten\_tk,

so\_tk = @so\_tk,

ngan\_hang = @ngan\_hang,

trang\_thai = @trang\_thai

WHERE ma\_tknh = @ma\_tknh;

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**i. Transaction vô hiệu hóa tài khoản ngân hàng**

CREATE PROCEDURE SP\_UpdateTaiKhoanNH

@ma\_tknh INT

AS BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM TaiKhoanNH WHERE ma\_tknh = @ma\_tknh)

BEGIN

RAISERROR(N'Tài khoản không tồn tại!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE TaiKhoanNH

SET trang\_thai = 'inactive'

WHERE ma\_tknh = @ma\_tknh;

SELECT @@ROWCOUNT as so\_dong\_cap\_nhat;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

**j. Transaction thêm nhân viên và tài khoản nhân viên**

CREATE OR ALTER PROCEDURE dbo.SP\_ThemNhanVien

@HoTen NVARCHAR(100),

@Email NVARCHAR(100),

@Sdt VARCHAR(20) = NULL,

@Username VARCHAR(50),

@Password VARCHAR(255),

@VaiTro NVARCHAR(30),

@Provision BIT = 1

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRAN;

BEGIN TRY

IF EXISTS (SELECT 1 FROM dbo.TaiKhoan WHERE username = @Username)

BEGIN

RAISERROR(N'Username đã tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

IF EXISTS (SELECT 1 FROM dbo.NhanVien WHERE email = @Email)

BEGIN

RAISERROR(N'Email đã tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

INSERT INTO dbo.TaiKhoan(username, [password], trang\_thai)

VALUES (@Username, @Password, 'active');

DECLARE @ma\_tk INT = SCOPE\_IDENTITY();

INSERT INTO dbo.NhanVien(ho\_ten, email, sdt, ma\_tk)

VALUES (@HoTen, @Email, @Sdt, @ma\_tk);

DECLARE @ma\_nv INT = SCOPE\_IDENTITY();

IF (@VaiTro = N'TRUONG\_PHONG\_TC')

INSERT INTO dbo.TruongPhongTC(ma\_nv, quyen\_han) VALUES (@ma\_nv, N'Duyệt giao dịch');

ELSE IF (@VaiTro = N'NHAN\_VIEN\_TC')

INSERT INTO dbo.NhanVienTC(ma\_nv, chuyen\_mon, cap\_bac) VALUES (@ma\_nv, N'Tài chính', 'NhanVien');

ELSE IF (@VaiTro = N'KE\_TOAN')

INSERT INTO dbo.KeToan(ma\_nv, chung\_chi, kinh\_nghiem) VALUES (@ma\_nv, N'CPA', 0);

ELSE

BEGIN

RAISERROR(N'Vai trò không hợp lệ.', 16, 1);

ROLLBACK TRAN; RETURN;

END

IF (@Provision = 1)

EXEC dbo.SP\_ProvisionSqlAccount @Username, @Password;

COMMIT TRAN;

SELECT @ma\_nv AS ma\_nv\_moi;

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

THROW;

END CATCH

END;

**k. Transaction xóa nhân viên và tài khoản nhân viên**

CREATE OR ALTER PROCEDURE dbo.SP\_XoaNhanVien

@MaNv INT,

@HardDelete BIT = 0,

@DropSql BIT = 1

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRAN;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM dbo.NhanVien WHERE ma\_nv = @MaNv)

BEGIN

RAISERROR(N'Nhân viên không tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

DECLARE @username VARCHAR(50), @ma\_tk INT;

SELECT @ma\_tk = nv.ma\_tk, @username = tk.username

FROM dbo.NhanVien nv

JOIN dbo.TaiKhoan tk ON tk.ma\_tk = nv.ma\_tk

WHERE nv.ma\_nv = @MaNv;

IF (@HardDelete = 1)

BEGIN

IF EXISTS (SELECT 1 FROM dbo.GiaoDich WHERE ma\_nv\_tao\_nvtc = @MaNv OR ma\_nv\_duyet\_tp = @MaNv)

BEGIN

RAISERROR(N'Không thể xóa vĩnh viễn do đang được tham chiếu trong giao dịch.', 16, 1);

ROLLBACK TRAN; RETURN;

END

DELETE FROM dbo.NhanVien WHERE ma\_nv = @MaNv;

DELETE FROM dbo.TaiKhoan WHERE ma\_tk = @ma\_tk;

IF (@DropSql = 1 AND @username IS NOT NULL)

EXEC dbo.SP\_DropSqlAccount @username;

END

ELSE

BEGIN

UPDATE dbo.TaiKhoan SET trang\_thai = 'inactive' WHERE ma\_tk = @ma\_tk;

IF (@DropSql = 1 AND @username IS NOT NULL)

EXEC dbo.SP\_DropSqlAccount @username;

END

COMMIT TRAN;

SELECT 1 AS success;

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

THROW;

END CATCH

END;

**l. Transaction kích hoạt lại tài khoản của nhân viên**

CREATE OR ALTER PROCEDURE dbo.SP\_KichHoatNhanVien

@MaNv INT,

@Provision BIT = 1

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRAN;

BEGIN TRY

IF NOT EXISTS (SELECT 1 FROM dbo.NhanVien WHERE ma\_nv = @MaNv)

BEGIN

RAISERROR(N'Nhân viên không tồn tại.', 16, 1);

ROLLBACK TRAN; RETURN;

END

DECLARE @ma\_tk INT, @username VARCHAR(50), @password VARCHAR(255);

SELECT @ma\_tk = nv.ma\_tk, @username = tk.username, @password = tk.[password]

FROM dbo.NhanVien nv

JOIN dbo.TaiKhoan tk ON tk.ma\_tk = nv.ma\_tk

WHERE nv.ma\_nv = @MaNv;

UPDATE dbo.TaiKhoan SET trang\_thai = 'active' WHERE ma\_tk = @ma\_tk;

IF (@Provision = 1 AND @username IS NOT NULL AND @password IS NOT NULL)

EXEC dbo.SP\_ProvisionSqlAccount @username = @username, @password = @password;

COMMIT TRAN;

SELECT 1 AS success;

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0 ROLLBACK TRAN;

THROW;

END CATCH

END;

**m. Transaction khi thực hiện thay đổi trạng thái không đúng đối với giao dịch**

CREATE OR ALTER TRIGGER TR\_GiaoDich\_ValidateTransition

ON dbo.GiaoDich

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

IF EXISTS (

SELECT 1

FROM inserted i

JOIN deleted d ON d.ma\_gd = i.ma\_gd

WHERE d.trang\_thai IN ('DA\_DUYET','TU\_CHOI')

AND i.trang\_thai <> d.trang\_thai

)

BEGIN

RAISERROR(N'Không được phép thay đổi trạng thái sau khi đã duyệt hoặc từ chối.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

IF EXISTS (

SELECT 1

FROM inserted i

JOIN deleted d ON d.ma\_gd = i.ma\_gd

WHERE d.trang\_thai = 'CHO\_DUYET'

AND i.trang\_thai NOT IN ('CHO\_DUYET','DA\_DUYET','TU\_CHOI')

)

BEGIN

RAISERROR(N'Trạng thái không hợp lệ. CHO\_DUYET chỉ được chuyển sang DA\_DUYET hoặc TU\_CHOI.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

END;

**n. Transaction khi thực hiện giao dịch với dự án, ngân hàng không hoạt động**

CREATE OR ALTER TRIGGER TR\_GiaoDich\_ValidateFKActive ON dbo.GiaoDich

AFTER INSERT, UPDATE

AS

BEGIN

SET NOCOUNT ON;

IF EXISTS (

SELECT 1

FROM inserted i

JOIN dbo.TaiKhoanNH tknh ON tknh.ma\_tknh = i.ma\_tknh

WHERE tknh.trang\_thai <> 'active'

) BEGIN

RAISERROR(N'Tài khoản ngân hàng đã inactive. Không thể tạo/cập nhật giao dịch.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

IF EXISTS (

SELECT 1

FROM inserted i

JOIN dbo.DuAn da ON da.ma\_du\_an = i.ma\_du\_an

WHERE i.ma\_du\_an IS NOT NULL

AND da.trang\_thai <> 'active'

) BEGIN

RAISERROR(N'Dự án đã inactive/hoàn thành. Không thể tạo/cập nhật giao dịch gắn dự án này.', 16, 1);

ROLLBACK TRANSACTION; RETURN;

END

END;

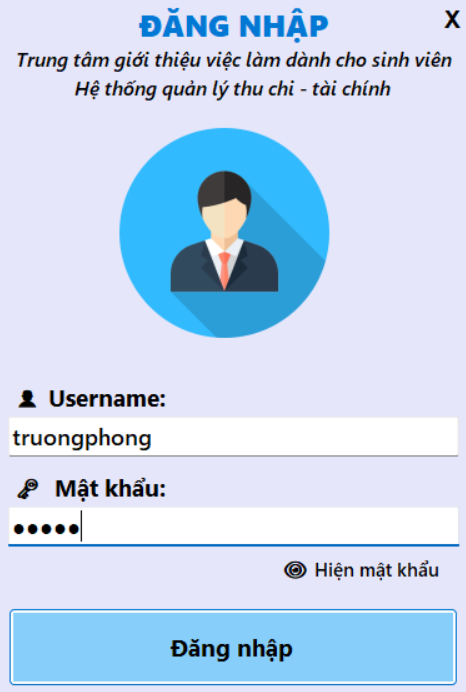
# **CHƯƠNG 6: THIẾT KẾ GIAO DIỆN**

## **1. Môi trường cài đặt và công nghệ sử dụng**

- Hệ cơ sở dữ liệu Microsoft SQL Server Management Studio 20.

- Chương trình Winform được viết bằng Microsoft Visual Studio 2022 - trên nền tảng ASP.NET Web Application (.NET Framework) C#.

## **2. Phần giao diện đăng nhập**

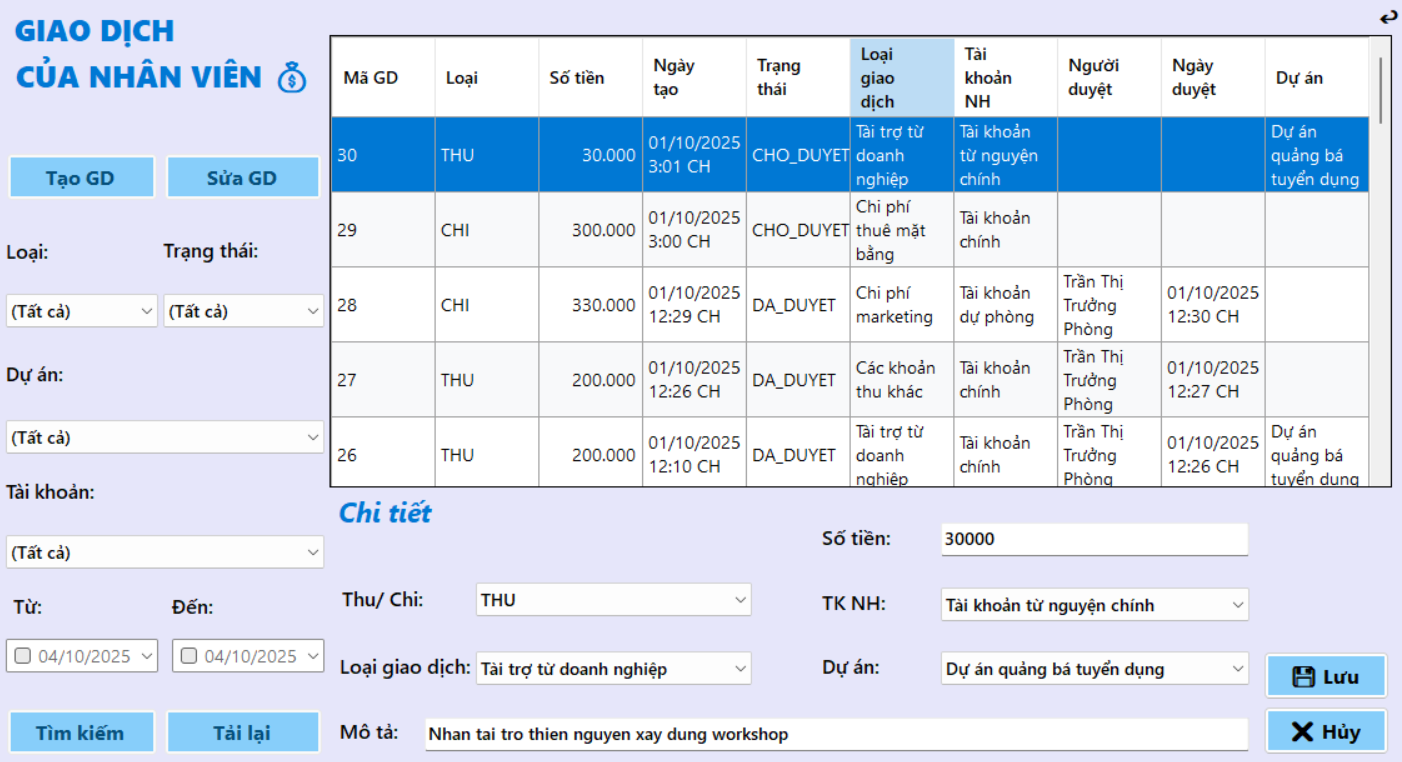


## **3. Phần giao diện của nhân viên tài chính**

**a. Trang chủ nhân viên tài chính**

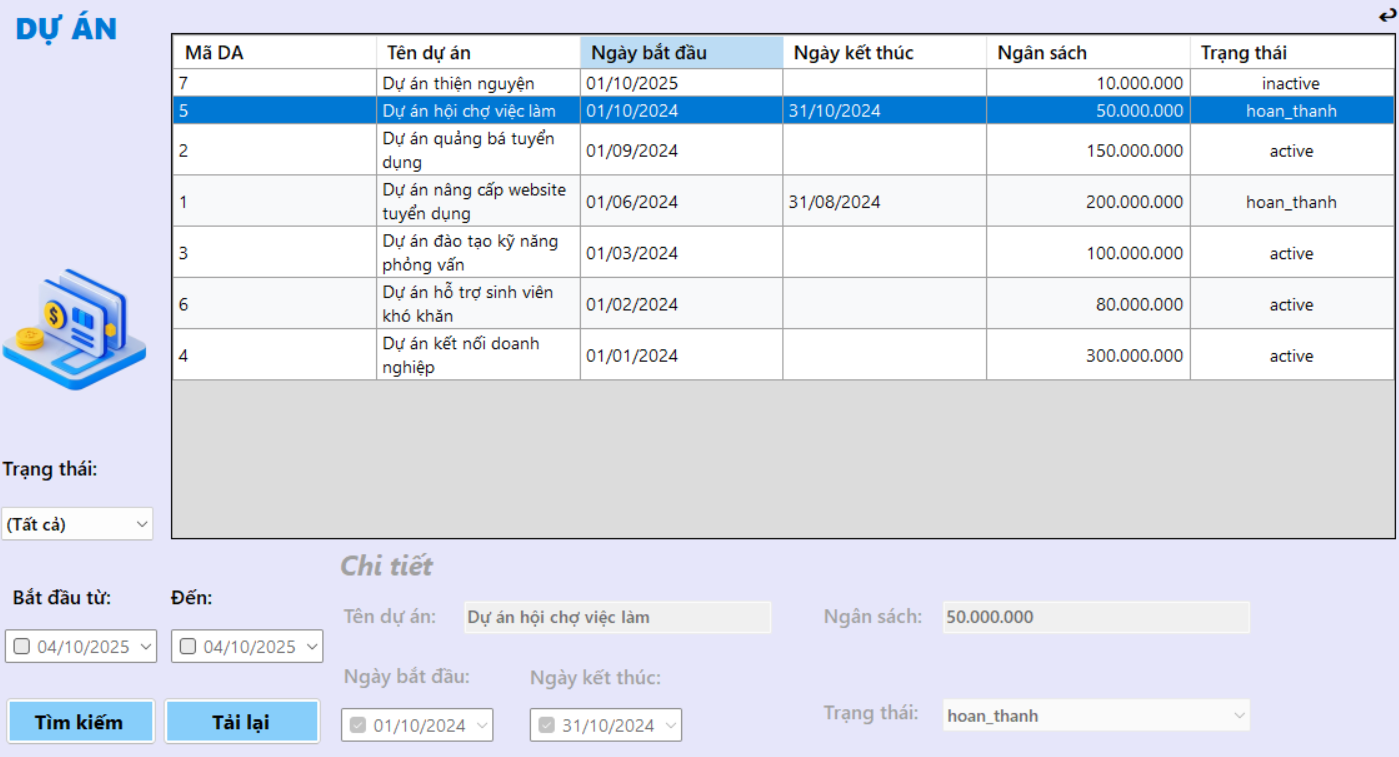


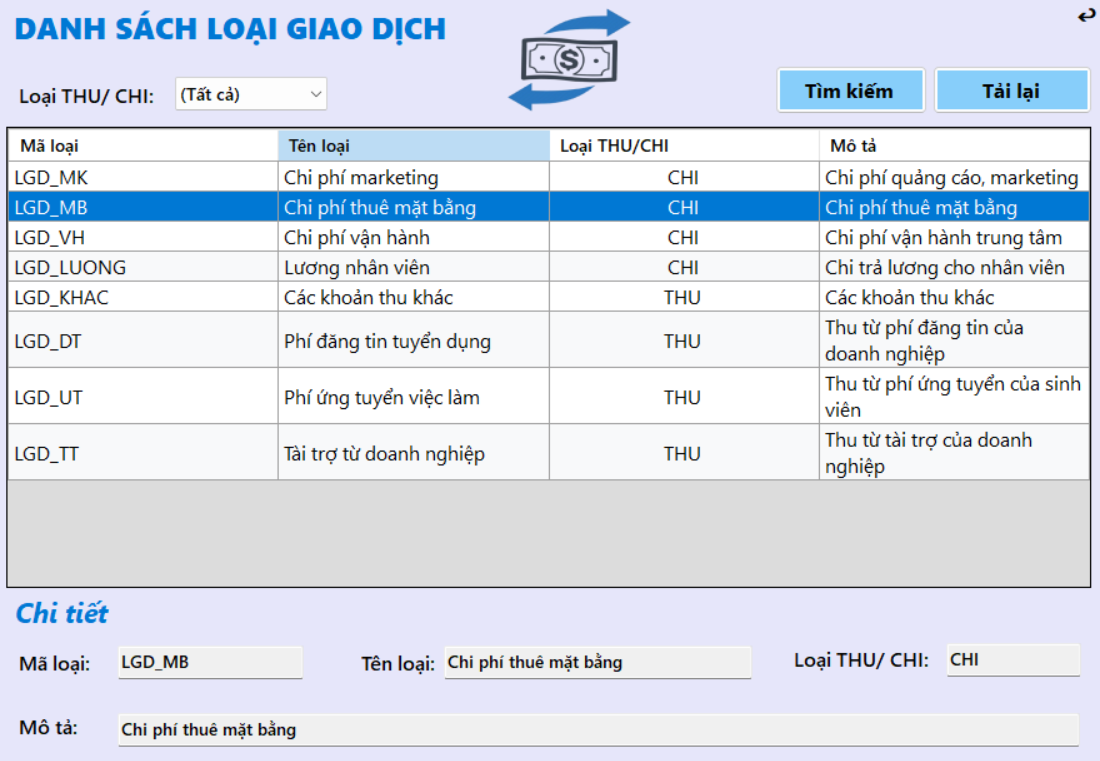
**b. Chức năng giao dịch của tôi**

****

**c. Các chức năng tra cứu (tài khoản ngân hàng, dự án, loại giao dịch)**

****

****

****

## **4. Phần giao diện của trưởng phòng tài chính**

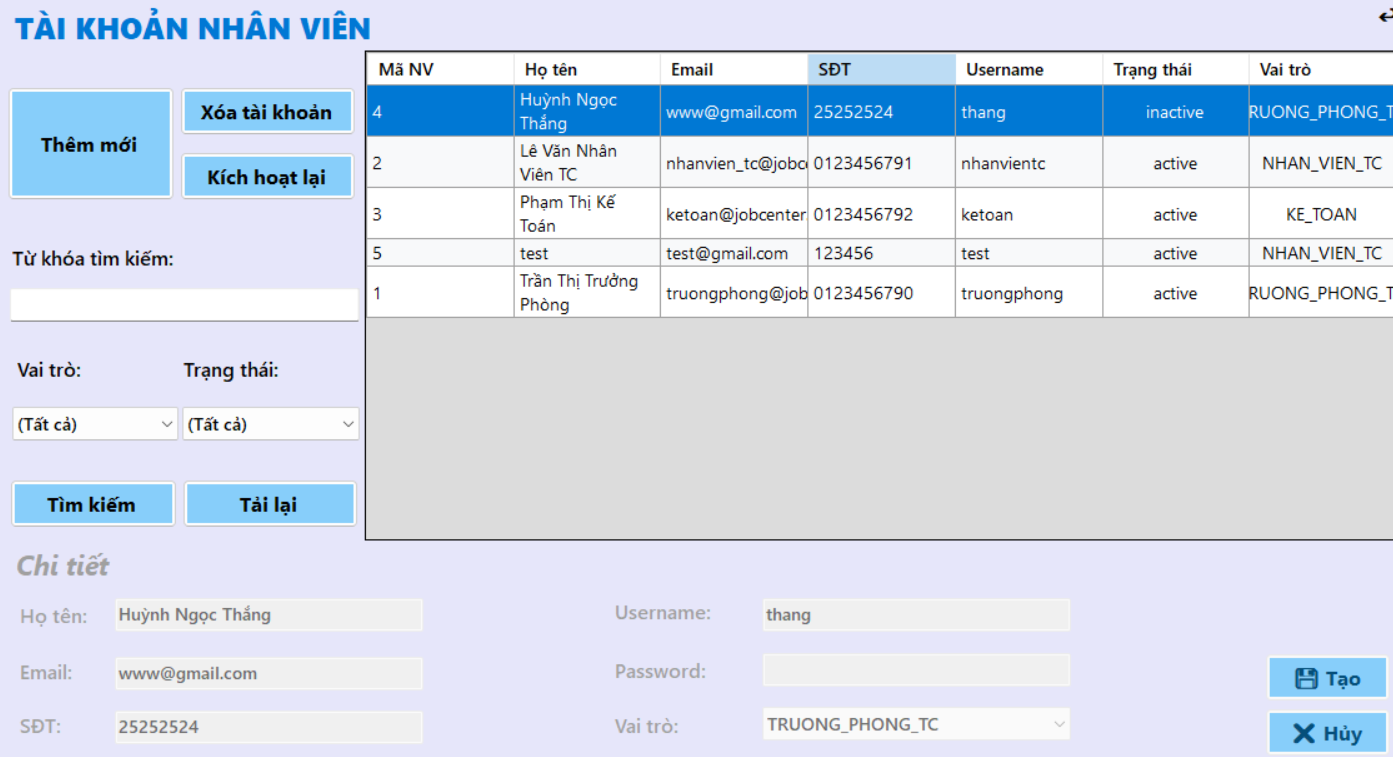
**a. Trang chủ trưởng phòng tài chính**



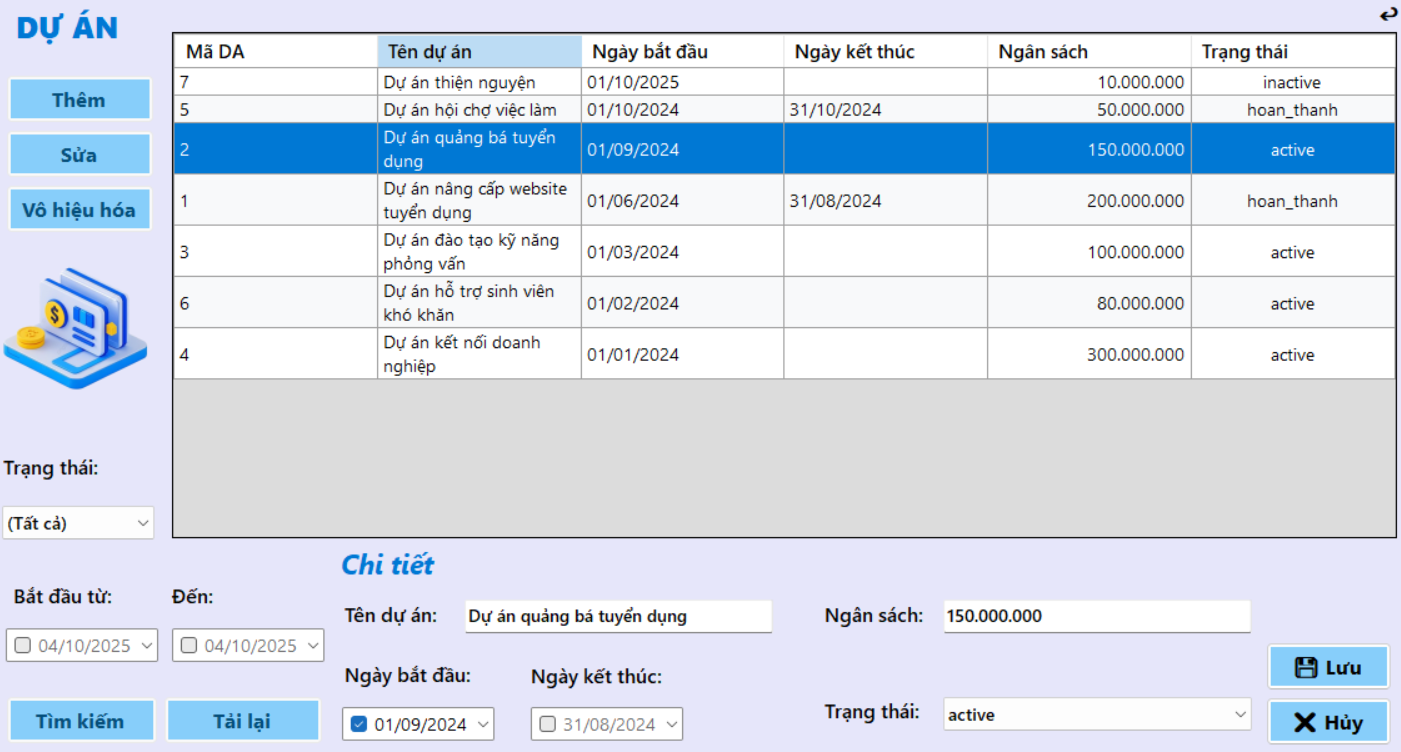
**b. Chức năng giao dịch chờ duyệt**

****

**c. Các chức năng quản lý (tài khoản nhân viên, tài khoản ngân hàng, dự án)**

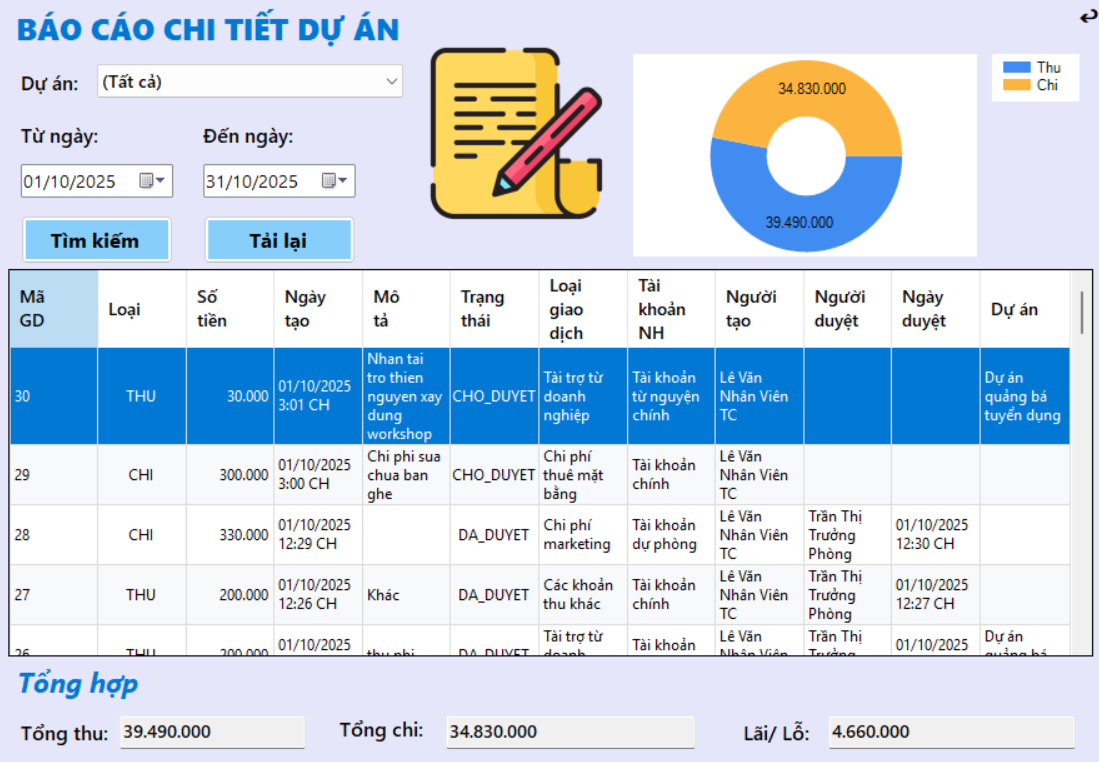
****

****

****

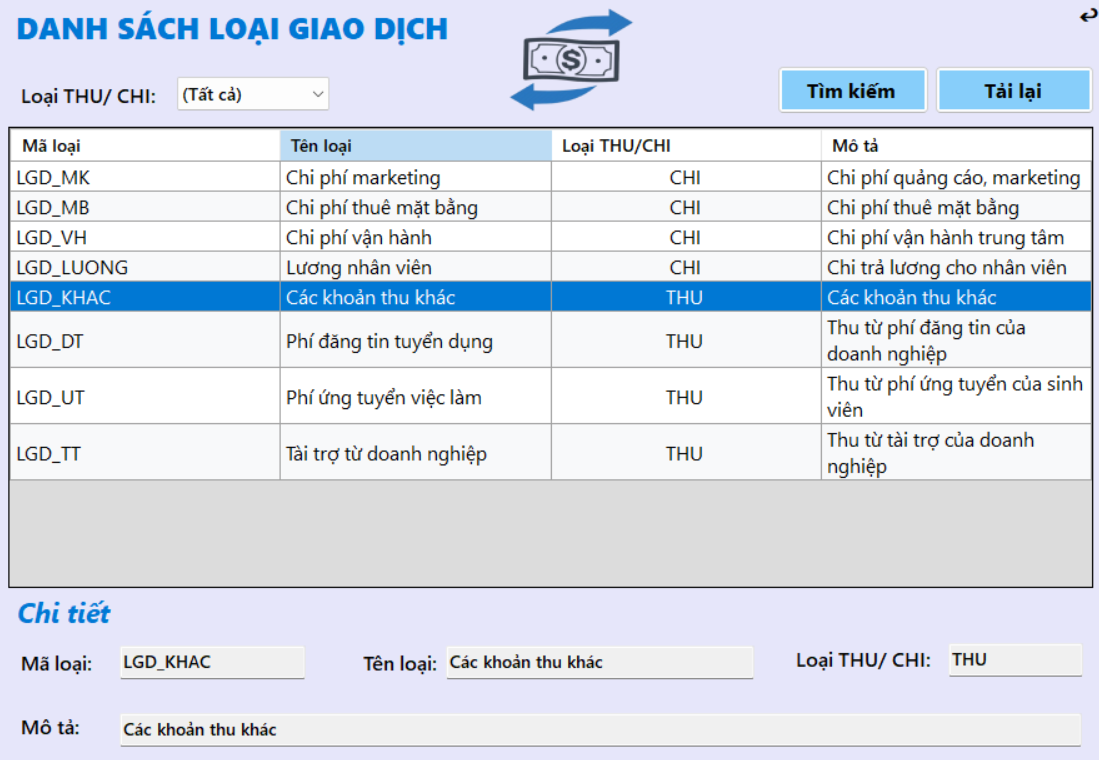
**d. Các chức năng báo cáo và thống kê**

****

****

**e. Các chức năng tra cứu (lịch sử giao dịch, loại giao dịch)**

****

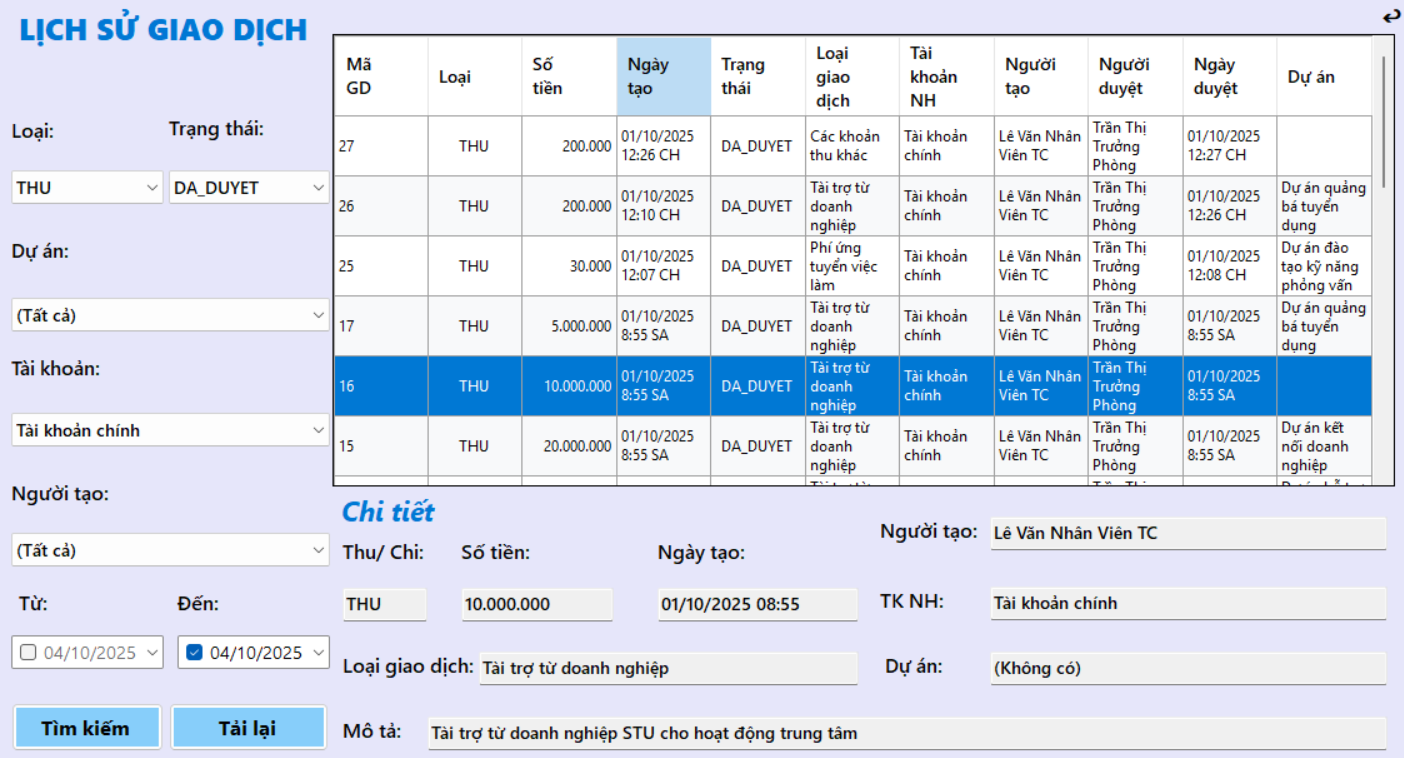
****

## **5. Phần giao diện của kế toán**

**a. Trang chủ của kế toán**

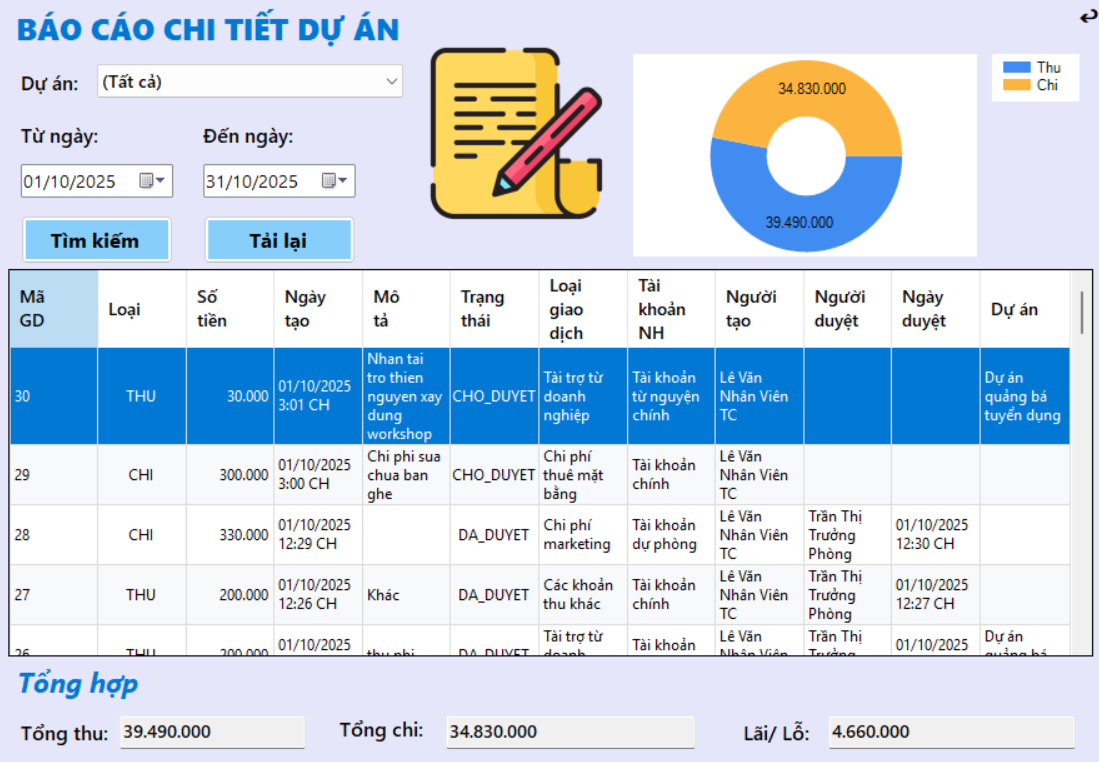
****

**b. Chức năng lịch sử giao dịch**

****

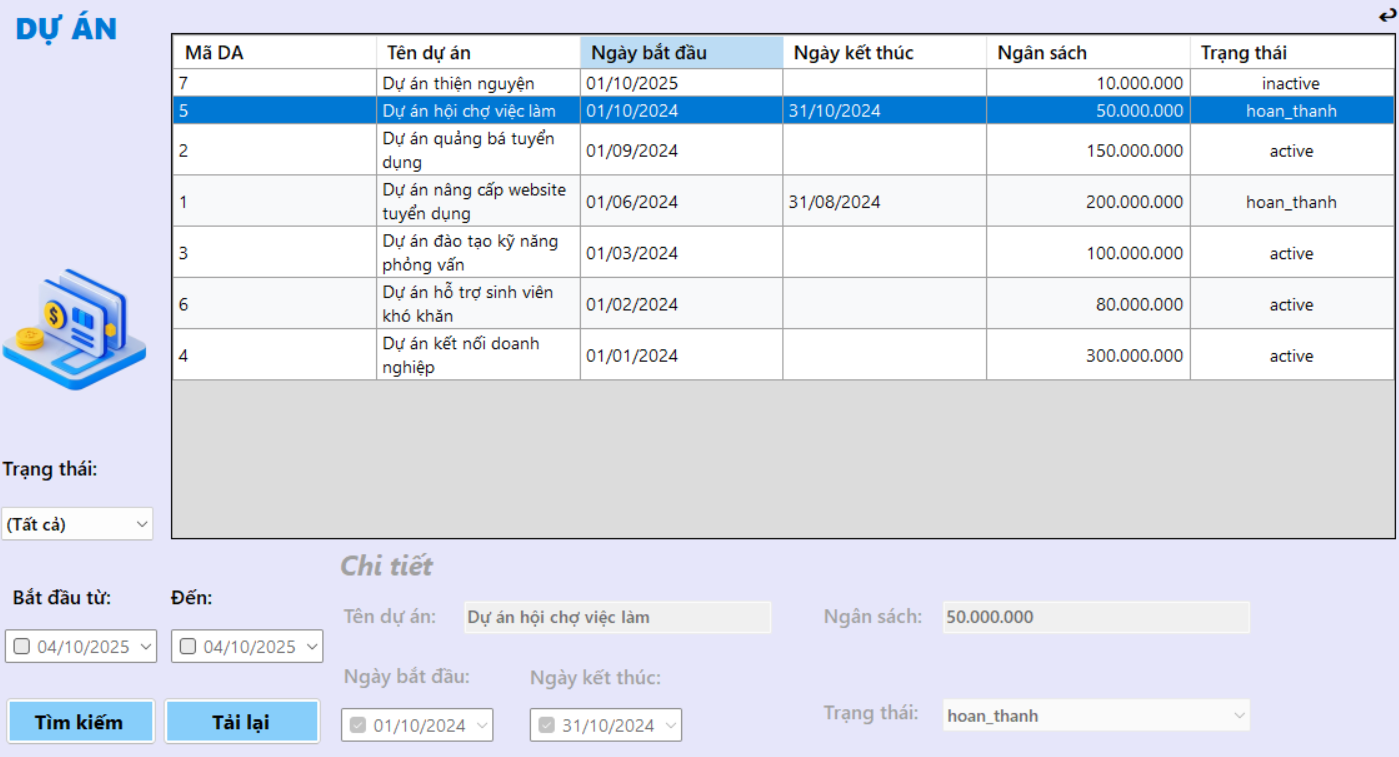
**c. Các chức năng báo cáo và thống kê**

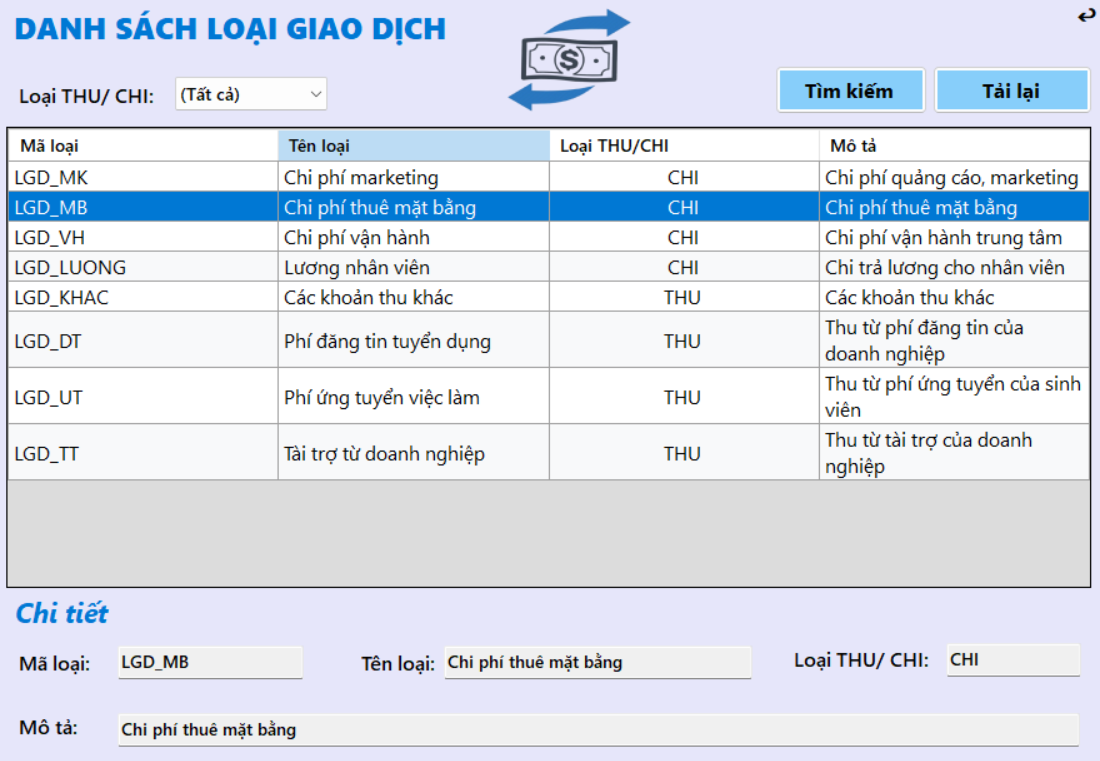
****

****

**d. Các chức năng tra cứu (tài khoản ngân hàng, dự án, loại giao dịch)**

****

****

****