TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────

ĐỒ ÁN

**PROJECT II**

NGÀNH CÔNG NGHỆ THÔNG TIN

**XÂY DỰNG FORM NHẬP LIỆU TRÊN MS OFFICE – KẾT NỐI VỚI CƠ SỞ DỮ LIỆU**

Sinh viên thực hiện: **Nguyễn Huy Định**

**MSSV 20161042**

Lớp CNTT1.01 - K61

Giáo viên hướng dẫn: ThS. **Nguyễn Đức Tiến**

HÀ NỘI 04-2018

# PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN MÔN HỌC

**1. Thông tin về sinh viên**

Họ và tên sinh viên: Nguyễn Huy Định

Điện thoại liên lạc: 0387625757 Email: huydinh3010@gmail.com

Lớp: CNTT1.01-K61 Hệ đào tạo: đại học chính quy

Thời gian làm đồ án: Từ ngày 24/1 đến ngày 1/5/2019

**2. Mục đích nội dung của đồ án**

* Thực hành làm việc với một hệ quản trị cơ sở dữ liệu trong quá trình thiết kế phần mềm.

**3. Các nhiệm vụ cụ thể của đồ án**

* Tìm hiểu về Git.
* Xây dựng form nhập liệu trên MS Excel:
  + Từ một file Excel của người dùng hãy lấy thông tin, kiểm tra thông tin sau đó đẩy thông tin vào cơ sở dữ liệu.
  + Lấy một số thông tin từ bảng của cơ sở dữ liệu và hiển thị trên một sheet, hiển thị trở lại form.
  + Kiểm soát các quyền của người dùng đối với form Excel,…
* Kiểm thử
* Tổng kết và đánh giá.

**4. Lời cam đoan của sinh viên:**

Tôi -Nguyễn Huy Định- cam kết đồ án là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của giáo viên. Các kết quả nêu trong đồ án là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày … tháng … năm …..*  Tác giả đồ án |

**5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của đồ án**

|  |  |
| --- | --- |
|  | *Hà Nội, ngày … tháng … năm …..*  Giáo viên hướng dẫn |

MỤC LỤC

[PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN MÔN HỌC 2](#_Toc6753019)

[DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ 6](#_Toc6753020)

[PHẦN 1: TÌM HIỂU VỀ GIT 7](#_Toc6753021)

[**I. Quản lý phiên bản với GIT 7**](#_Toc6753022)

[**II. Cơ bản về Git 8**](#_Toc6753023)

[**2.1. Tạo một Repository 9**](#_Toc6753024)

[**2.2. Commit – ghi thay đổi vào repository 10**](#_Toc6753025)

[**2.3. Git Log và Undo Commit 11**](#_Toc6753026)

[**2.4. Đánh dấu commit với Tag 12**](#_Toc6753027)

[**III. Phân nhánh – Branch 12**](#_Toc6753028)

[**3.1. Branch, phân nhánh và gộp nhánh 12**](#_Toc6753029)

[**3.2. Các lệnh với Branch 16**](#_Toc6753030)

[**3.3. Remote Branch 16**](#_Toc6753031)

[PHẦN 2: Xây dựng form nhập liệu bằng Excel 18](#_Toc6753032)

[**I. Đặt vấn đề và định hướng giải pháp 18**](#_Toc6753033)

[**1.1 Đặt vấn đề 18**](#_Toc6753034)

[**1.2 Excel và các công cụ sử dụng 18**](#_Toc6753035)

[**II. Phân tích và thiết kế các chức năng của hệ thống 20**](#_Toc6753036)

[**2.1. Các chức năng 20**](#_Toc6753037)

[**2.2. Use case 21**](#_Toc6753038)

[**2.3. Xây dựng cơ sở dữ liệu 22**](#_Toc6753039)

[**2.3.1. Thiết kế cơ sở dữ liệu 22**](#_Toc6753040)

[**2.3.2. Kết nối đến SQL Server 23**](#_Toc6753041)

[**2.4. Thiết kế các chức năng 24**](#_Toc6753042)

[**2.4.1. Form nhập liệu 24**](#_Toc6753043)

[**2.4.2. Kết nối và đưa dữ liệu lên cơ sở dữ liệu 26**](#_Toc6753044)

[**2.4.3. Lấy dữ liệu từ cơ sở dữ liệu hiển thị lên sheet History 27**](#_Toc6753045)

[**2.4.4. Lấy dữ liệu từ cơ sở dữ liệu hiển thị lên form 28**](#_Toc6753046)

[**III. KẾT LUẬN 29**](#_Toc6753047)

[TÀI LIỆU THAM KHẢO 30](#_Toc6753048)

DANH MỤC BẢNG

[*Bảng 1: Bảng XetNghiem trong CSDL 22*](#_Toc6755178)

[*Bảng 2: Bảng KetQuaXetNghiem trong CSDL 23*](#_Toc6755179)

DANH MỤC HÌNH VẼ

[*Hình 1: Mô hình hoạt động của DVCS 7*](#_Toc6754570)

[*Hình 2: Thao tác với dữ liệu trên Git 8*](#_Toc6754571)

[*Hình 3: Tiến trình công việc của Git 9*](#_Toc6754572)

[*Hình 4: Trạng thái của tập tin 10*](#_Toc6754573)

[*Hình 5: Nhánh master 13*](#_Toc6754574)

[*Hình 6: con trỏ HEAD 13*](#_Toc6754575)

[*Hình 7: Làm việc với các nhánh 14*](#_Toc6754576)

[*Hình 8: Trộn nhánh master và hotfix 14*](#_Toc6754577)

[*Hình 9: Trộn nhánh master và iss53 15*](#_Toc6754578)

[*Hình 10: Remote Branch 16*](#_Toc6754579)

[*Hình 11: Đồng bộ các thay đổi với máy chủ 17*](#_Toc6754580)

[*Hình 12: File xét nghiệm 19*](#_Toc6754581)

[*Hình 13: Biểu đồ phân rã chức năng 20*](#_Toc6754582)

[*Hình 14: Biểu đồ usecase 21*](#_Toc6754583)

[*Hình 15: Sơ đồ thực thể quan hệ 22*](#_Toc6754584)

[*Hình 16: Data Validation của mã số cuộc hẹn 25*](#_Toc6754585)

[*Hình 17: Tùy chọn trong Protect Sheet 25*](#_Toc6754586)

[*Hình 18: Biểu đồ triển khai btnSubmit\_Click 26*](#_Toc6754587)

[*Hình 19: Power Query Editor 27*](#_Toc6754588)

[*Hình 20: Thiết lập Query Properties 27*](#_Toc6754589)

[*Hình 21: Sử dụng Label để hiển thị 28*](#_Toc6754590)

[*Hình 22: Thiết lập in các ActiveX Controls. 28*](#_Toc6754591)

[*Hình 23: Giao diện tương tác với người dùng của chương trình 29*](#_Toc6754592)

# DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ

|  |  |  |
| --- | --- | --- |
| **Số thứ tự** | **Từ viết tắt** | **Ý nghĩa** |
| 1 | CSDL | Cơ sở dữ liệu |
| 2 | DVCS | Distributed Version Control System – hệ thống quản lý phiên bản phân tán |
| 3 | ODBC | Open Database Connectivity – là một phương pháp truy xuất CSDL chuẩn |
| 4 | VBA | Visual Basic for Application |
| 5 | VB | Visual Basic |
| 6 | MS Office | Microsoft Office  Contents  [PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN MÔN HỌC 2](#_Toc6752189)  [DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ 6](#_Toc6752190)  [PHẦN 1: TÌM HIỂU VỀ GIT 7](#_Toc6752191)  [I. Quản lý phiên bản với GIT 7](#_Toc6752192)  [II. Cơ bản về Git 8](#_Toc6752193)  [III. Phân nhánh – Branch 12](#_Toc6752194)  [PHẦN 2: Xây dựng form nhập liệu bằng Excel 18](#_Toc6752195)  [I. Đặt vấn đề và định hướng giải pháp 18](#_Toc6752196)  [1.1. Đặt vấn đề 18](#_Toc6752197)  [1.2. Excel và các công cụ sử dụng 18](#_Toc6752198)  [II. Phân tích và thiết kế các chức năng của hệ thống 20](#_Toc6752199)  [2.1. Các chức năng 20](#_Toc6752200)  [2.2. Usecase 21](#_Toc6752201)  [2.3. Xây dựng cơ sở dữ liệu 22](#_Toc6752202)  [2.3.1. Thiết kế cơ sở dữ liệu 22](#_Toc6752203)  [2.3.2. Kết nối đến SQL Server 23](#_Toc6752204)  [2.4. Thiết kế chức năng 24](#_Toc6752205)  [2.4.1. Form nhập liệu 24](#_Toc6752206)  [2.4.2. Kết nối và đưa dữ liệu lên cơ sở dữ liệu 26](#_Toc6752207)  [2.4.3. Lấy dữ liệu từ cơ sở dữ liệu đưa vào sheet 27](#_Toc6752208)  [2.4.4. Lấy thông tin từ cơ sở dữ liệu đưa vào form 28](#_Toc6752209)  [III. KẾT LUẬN 29](#_Toc6752210)  [TÀI LIỆU THAM KHẢO 30](#_Toc6752211) |

# PHẦN 1: TÌM HIỂU VỀ GIT

## Quản lý phiên bản với GIT

**Git** là tên gọi của một **hệ thống quản lý phiên bản phân tán** (Distributed Version Control System – ***DVCS***) là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay. DVCS nghĩa là hệ thống giúp mỗi máy tính có thể lưu trữ nhiều phiên bản khác nhau của một mã nguồn được nhân bản (***clone***) từ một kho chứa mã nguồn (***repository***), mỗi thay đổi vào mã nguồn trên máy tính sẽ có thể ủy thác (***commit***) rồi đưa lên máy chủ nơi đặt kho chứa chính. Và một máy tính khác (nếu họ có quyền truy cập) cũng có thể clone lại mã nguồn từ kho chứa hoặc clone lại một tập hợp các thay đổi mới nhất trên máy tính kia. Trong Git, thư mục làm việc trên máy tính gọi là **Working Tree**.



Hình : Mô hình hoạt động của DVCS

Cơ chế lưu trữ các phiên bản của Git là nó sẽ tạo ra một “ảnh chụp” (*snapshot*) trên mỗi tập tin và thư mục sau khi commit. Mỗi lần commit, Git sẽ “chụp lại” nội dung của tất cả các tập tin tại thời điểm đó và tạo ra một tham chiếu tới “ảnh” đó. Nếu như tập tin không có sự thay đổi nào, Git không lưu trữ tập tin đó lại một lần nữa mà chỉ tạo một liên kết tới tập tin gốc đã tồn tại trước đó.



Hình : Thao tác với dữ liệu trên Git

## Cơ bản về Git

Mỗi tập tin trong Git đều được quản lý theo ba trạng thái: committed, modified và staged. Committed có nghĩa là dữ liệu đã được lưu trữ một cách an toàn trong cơ sở dữ liệu. Modified có nghĩa là tập tin đã bị thay đổi nhưng chưa commit vào cơ sở dữ liệu. Staged nghĩa là tập tin đã được đánh dấu sẽ commit phiên bản hiện tại trong lần commit sắp tới. Điều này tạo ra ba phần riêng biệt của một dự án sử dụng Git: thư mục Git (*git directory – repository*), thư mục làm việc (*working directory*) và khu vực tổ chức (*staging* *area*).

* Git Directory (*Repository*) là nơi Git lưu trữ các metadata và cơ sở dữ liệu về dự án (giống như một kho chứa). Đây là phần quan trọng nhất của Git, nó là phần được sao lưu về khi tạo một bản sao (*clone*) của một kho chứa (*repository*) từ một máy tính khác.
* Working Directory là bản sao một phiên bản của dự án. Những tập tin này được kéo về (*pulled*) từ cơ sở dữ liệu được nén lại trong thư mục Git và lưu trên máy tính của người làm việc.
* Staging Area là một khu vực mà nó sẽ được chuẩn bị cho quá trình commit, khu vực này sẽ lưu trữ những thay đổi trên tập tin để nó có thể được commit, vì muốn commit tập tin nào thì tập tin đó phải nằm trong Staging Area.

Tiến trình công việc (*workflow*) cơ bản của Git:

* Thay đổi các tập tin trong thư mục làm việc.
* Tổ chức các tập tin, tạo mới “ảnh” của các tập tin đó vào Staging Area.
* Commit: “ảnh” của các tập tin trong Staging Area được lưu vào Git Directory.



Hình : Tiến trình công việc của Git

* 1. **Tạo một Repository**

Repository là nơi lưu trữ mã nguồn và một số người khác có thể clone lại mã nguồn đó nhằm làm việc. Repository có hai loại là Local Repository (kho chứa trên máy cá nhân) và Remote Repository (kho chứa trên máy chủ từ xa).

Lệnh tạo một repository, cần cd đến thư mục lưu repository ví dụ:

$ git init git\_example

Initialized empty Git repository in C:\git\_example\.git\

Ngoài ra có thể clone repository từ một máy chủ từ xa vào thư mục hiện tại bằng lệnh:

$ git clone <https://github.com/huydinh3010/Project-2.git>

Hoặc đổi tên thư mục lưu repository bằng việc thêm tham số tên thư mục:

$ git clone <https://github.com/huydinh3010/Project-2.git> my\_repo

Url ở trên lấy từ máy chủ lưu trữ repository: github.com. Ngoài ra, ta có thể tương tác với máy chủ github.com như tạo repository, tạo nhánh, quản lý mã nguồn trên máy chủ,…

* 1. **Commit – ghi thay đổi vào repository**

Commit là hành động để Git lưu lại một bản chụp (*snapshot*) của các thay đổi trong thư mục làm việc, các tập tin và thư mục được thay đổi phải nằm trong Staging Area. Mỗi lần commit, Git sẽ được lưu lại lịch sử chỉnh sửa của mã nguồn kèm theo tên và địa chỉ email của người commit. Ngoài ra Git cũng cho phép khôi phục lại tập tin trong lịch sử commit của nó để chia cho một phân nhánh (*branch*) khác.

Lệnh commit trong Git:

$ git commit -m “lời nhắn”

Điều kiện để commit một tập tin là tập tin đó cần phải được đưa vào trạng thái tracked bằng lệnh

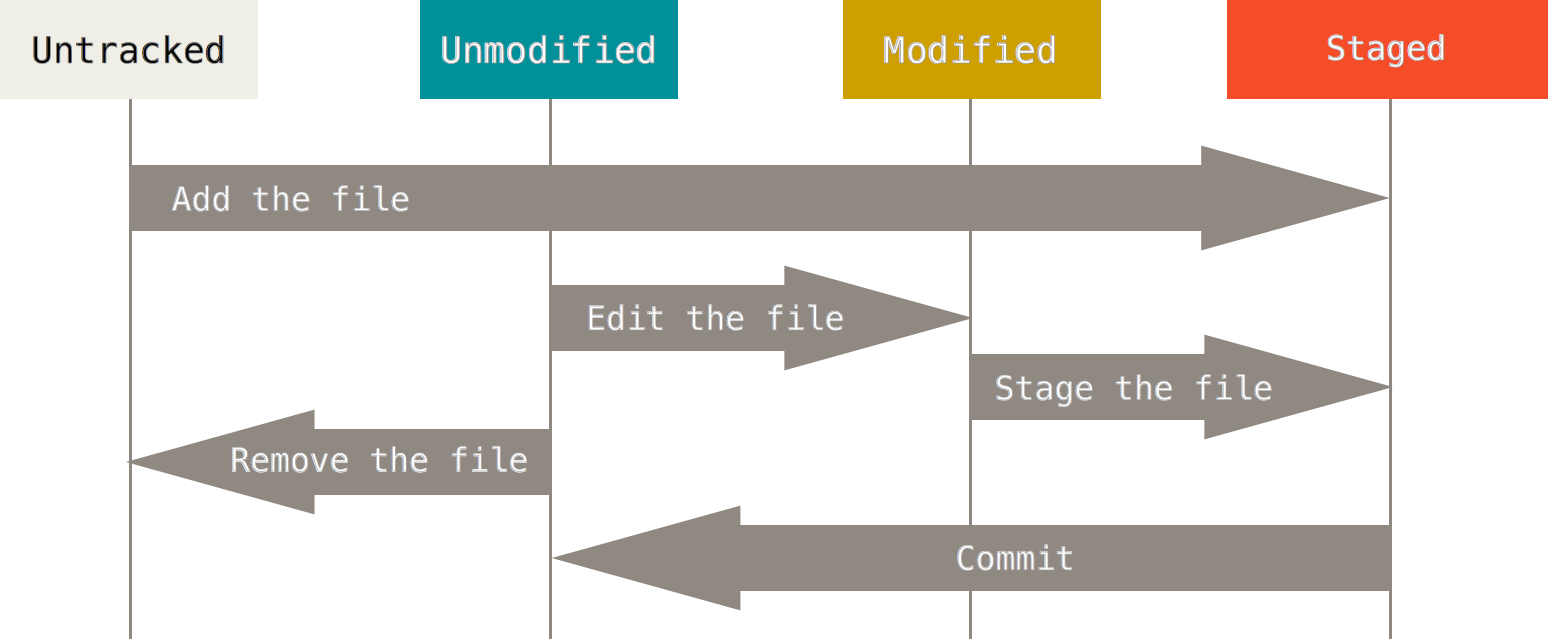
$ git add tên\_file

Trong Git có hai loại trạng thái chính trong thư mục làm việc đó là Tracked và Untracked, cụ thể như sau:

* Tracked: tập tin đã được đánh dấu theo dõi trong Git để ta làm việc với nó. Trạng thái này có ba trạng thái phụ khác là Unmodified (chưa chỉnh sửa), Modified (đã chỉnh sửa) và Staged (sẵn sàng để commit).
* Untracked: các tập tin còn lại – bất kỳ tập tin nào trong thư mục làm việc mà không có ảnh lần commit trước hoặc không ở trong Staging Area. Nếu tạo hoặc them mới tập tin thì tập tin đó sẽ ở trạng thái này.

Nếu tập tin đã được Tracked nhưng đang rơi vào trạng thái Modified thì nó vẫn không thể commit được mà phải đưa tập tin về trạng thái Staged bằng lệnh “*git add”*. Tóm lại một tập tin sau khi thay đổi hoặc tạo mới thì nó phải được thêm vào Staging Area với lệnh “*git add”.* Tuy nhiên ta có thể đưa một tập tin đã được Tracked để commit mà không cần đưa nó vào Staging Area với tham số *-a* trong lệnh “*git commit”.* Ví dụ:

$ git commit -a -m “Bỏ qua Staging Area”



Hình : Trạng thái của tập tin

Để xem trạng thái của tập tin trong thư mục làm việc ta sử dụng lệnh:

$ git status

Để đưa tập tin từ Tracked về Untracked ta dùng lệnh “*rm”*, lệnh này sẽ đưa tập tin về trạng thái Untracked nhưng không xóa hẳn trong ổ cứng. Ví dụ:

$ rm tên\_file

Để xóa hẳn tập tin khỏi ổ cứng ta dùng lệnh:

$ git rm -f tên\_file

* 1. **Git Log và Undo Commit**

Một việc thường xuyên làm trong Git nếu làm việc nhóm đó là kiểm tra xem những ai đã commit vào dự án đang làm việc, cũng như cách để undo lần commit trước đó nếu như cảm thấy thiếu một tập tin nào đó hoặc có sai sót trong phiên bản hiện tại để bổ sung và chỉnh sửa.

Để xem lịch sử các lần commit trước đó ta sử dụng lệnh:

$ git log [-p] [-1]

Mỗi commit sẽ chứa một mã check sum được băm trước khi lưu dữ liệu và kết quả của lệnh log trong mỗi lần commit sẽ có thông tin của mã này. Cùng với đó là tên người commit, địa chỉ email, ngày lưu và lời nhắn.

Các tham số cho lệnh:

* -p: hiển thị chi tiết.
* -1: hiển thị lần commit gần nhất.
* --since, --after: xem lần commit kể từ ngày nhất định.
* --until: xem các lần commit trước từ ngày nhất định.
* --author: xem lần commit của một người nào đó.
* --grep: lọc các chuỗi trong log và in ra
* --pretty: sử dụng để lọc lời nhắn, lọc mã băm,…

Nếu cần xóa bỏ lệnh commit trước và cần undo để commit lại thì có thể sử dụng tham số --amend trong lệnh git commit. Ví dụ:

$ git commit –amend -m “lời nhắn”

Lưu ý rằng undo nghĩa là quay trở lại bước commit lần trước, do vậy nếu cần bổ sung tập tin nào vào để commit thì hãy đưa tập tin đó vào Staging Area trước. Nếu một tập tin đã bị đưa vào Staging Area mà muốn loại bỏ để không bị commit theo thì ta sử dụng lệnh sau, HEAD sẽ được giải thích trong phần phân nhánh.

$ git reset HEAD tên\_file

* 1. **Đánh dấu commit với Tag**

Khi dự án trở lên lớn hơn với những lần commit dày đặc sẽ gây khó khăn khi cần xem lại thông tin của lần commit trước. Việc đánh dấu (*tag*) cho mỗi commit sẽ giúp cho việc đối chiếu và quản lý được hiệu quả hơn.

Trong Git có hai kiểu Tag chính đó là:

* Lightweight Tag: các tag này chỉ đơn thuần là đánh dấu snapshot của commit.
* Annotated Tag: có thể đặt tiêu đề cho tag, thông tin về người tag, ngày tag,.. sẽ được hiển thị khi xem.

Thêm Lightweight Tag cho commit cuối cùng:

$ git tag tên\_tag

Xem danh sách các tag trong dự án:

$ git tag

Thêm Annotated Tag cho commit cuối cùng:

$ git tag -a tên\_tag -m “lời nhắn”

Show thông tin log của commit có tag = tên\_tag:

$ git show tên\_tag

Thêm tag cho các commit cũ bằng cách sử dụng checksum để định danh cho commit. Lấy check sum bằng tham số lọc “*--pretty=oneline”* của lệnh log sau đó truyền tham số checksum hoặc một vài giá trị đầu của checksum vào lệnh thêm tag, ví dụ:

$ git tag -a tên\_tag checksum -m “lời nhắn”

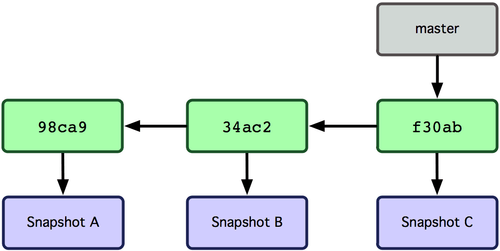
## Phân nhánh – Branch

* 1. **Branch, phân nhánh và gộp nhánh**

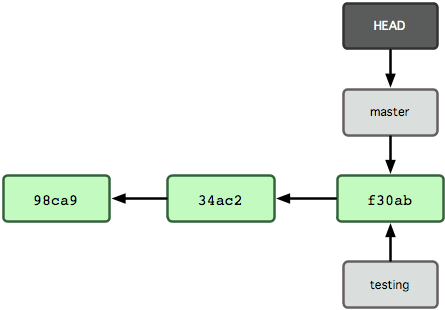
Khi thực hiện commit, Git lưu trữ đối tượng commit mà có chứa một con trỏ tới ảnh của nội dung, tác giả và thông điệp, và con trỏ trỏ tới commit cha. Commit đầu tiên không có cha, commit bình thường có một cha, và một commit là cha cho nhiều commit con chính là phân nhánh.

Một nhánh trong Git đơn thuần là một con trỏ có khả năng di chuyển được, trỏ đến một trong những commit này. Tên nhánh mặc định của Git là master. Trong những lần commit đầu tiên, chúng đều được trỏ đến nhánh master. Và sau mỗi lần commit nó sẽ tự động trỏ đến commit cuối cùng (theo hướng tiến lên).

Khi thực hiện tạo một nhánh mới, một con trỏ mới sẽ trỏ vào commit hiện tại của dự án. Git xác định nhánh đang làm việc bằng một con trỏ đặc biệt là con trỏ HEAD, con trỏ này sẽ trỏ đến nhánh đang làm việc. HEAD chỉ đơn giản chứa mã băm của commit mà nó trỏ tới.

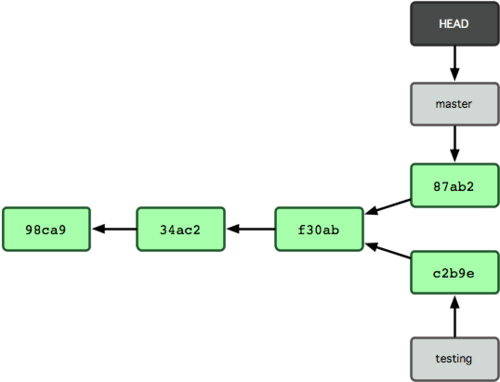


Hình : Nhánh master



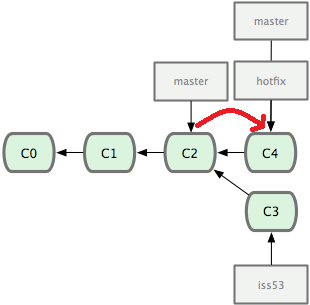
Hình : con trỏ HEAD

Tùy vị trí của HEAD mà khi ta thực hiện commit mới, commit này sẽ trỏ tới commit cha tương ứng trên nhánh đang làm việc. Lịch sử dự án bị tách ra theo các nhánh, sự thay đổi của các nhánh là cô lập không liên quan đến nhau, có thể chuyển qua lại giữa các nhánh và thực hiện gộp các nhánh khi cần thiết.



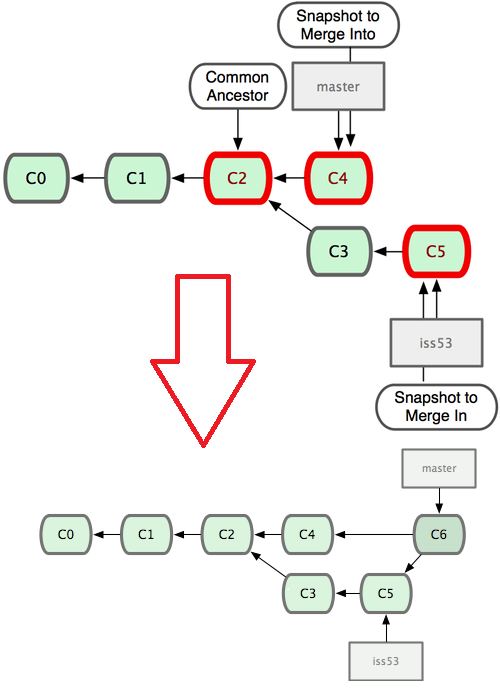
Hình : Làm việc với các nhánh

Sau khi hoàn thành vấn đề cần giải quyết trên nhánh mới, cần trộn nhánh lại với nhánh master. Tại đây Git thực hiện quá trình trộn nhánh. Nếu như commit hiện tại của các nhánh là cha con của nhau thì chỉ cần dịch chuyển con trỏ về phía trước mà không cần trộn – đây được gọi là *“fast forward”.*



Hình : Trộn nhánh master và hotfix

Trường hợp thứ hai là hai nhánh cần trộn không phải cha con của nhau. Git sẽ thực hiện trộn 3-chiều, sử dụng hai snapshot được trỏ tới bởi các nhánh và nút cha chung của cả hai. Sau đó Git tạo một snapshot mới – được hợp thành từ lần trộn 3-chiều này và cũng tự tạo một commit mới trỏ tới nó – gọi là commit trộn và đặc biệt là commit này có nhiều hơn một cha.



Hình : Trộn nhánh master và iss53

Quá trình trộn sẽ xảy ra lỗi nếu thay đổi cùng một nội dung của một tập tin ở hai nhánh cần trộn, Git sẽ báo lỗi và tạm dừng quá trình trộn và thêm các ký hiệu đánh dấu trong tập tin đó. Để giải quyết vấn đề này, ta cần phải chọn một trong hai phần khác nhau của tập tin trên hai nhánh và tự trộn chúng lại.

* 1. **Các lệnh với Branch**

Đọc tập tin *“.git/HEAD”* để biết vị trí con trỏ HEAD:

$ cat .git/HEAD

ref: refs/heads/master

Tạo một branch trỏ vào commit hiện tại, lưu ý con trỏ HEAD vẫn giữ nguyên branch hiện tại.

$ git branch tên\_branch

Checkout một branch: chuyển con trỏ HEAD qua branch khác để làm việc:

$ git checkout tên\_branch

Hiển thị danh sách nhánh:

$ git branch

Xem commit mới nhất trên từng nhánh:

$ git branch -v

Gộp dữ liệu từ một branch: chuyển dữ liệu từ một branch nào đó về branch đang trỏ đến (branch cần chuyển về đã được commit).

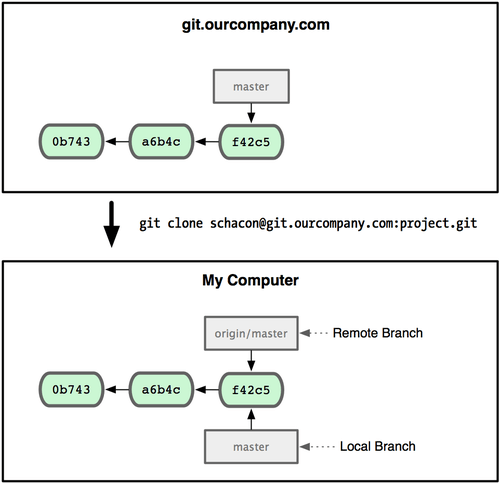
$ git merge tên\_branch

Xóa branch: cần phải gộp dữ liệu về master trước khi xóa.

$ git branch -d tên\_branch

* 1. **Remote Branch**

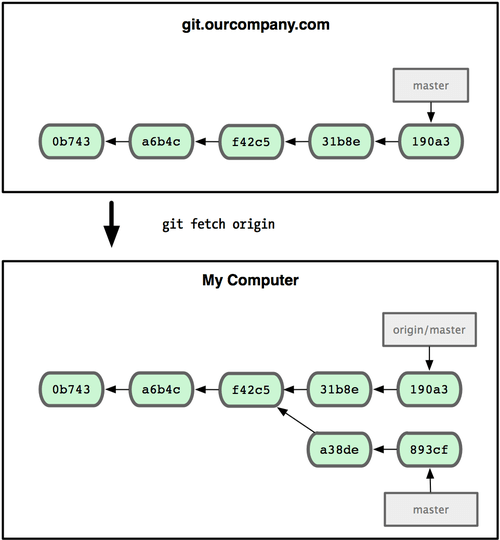
Nhánh từ xa (*remote branch*) là các tham chiếu tới trạng thái của các nhánh trên kho chứa của máy chủ. Khi tạo một bản sao từ một kho chứa trên máy chủ, Git sẽ tự động đặt tên là *origin* và tạo một con trỏ tới nhánh master (của dự án trên máy chủ) và đặt tên nó là origin/master trên máy tính (nội bộ). Đồng thời Git tạo một nhánh master riêng trên máy cục bộ để bắt đầu làm việc.



*Hình 10: Remote Branch*

Nếu ta thực hiện một số thay đổi trên nhánh master nội bộ, và cùng thời điểm đó, một người nào đó đẩy lên máy chủ và cập nhật nhánh master của nó thì trên máy cá nhân vẫn không thay đổi cho đến khi ta đồng bộ với máy chủ.

Để đồng bộ hóa các thay đổi, sử dụng lệnh *“git fetch origin”.* Lệnh này sẽ tìm kiếm máy chủ nào là origin và truy xuất toàn bộ dữ liệu chưa có từ nó và cập nhật cơ sở dữ liệu trên máy nội bộ, di chuyển con trỏ origin/master tới vị trí mới được cập nhật.



Hình : Đồng bộ các thay đổi với máy chủ

Đẩy một nhánh lên máy chủ và cập nhật vào nhánh cùng tên trên máy chủ:

$ git push origin tên\_nhánh

Tạo một nhánh trên máy chủ và đồng bộ với nhánh trên máy cục bộ:

$ git push origin tên\_nhánh\_cục\_bộ:tên\_nhánh\_trên\_máy\_chủ

Truy xuất nhánh trên máy chủ vào một nhánh cục bộ (mặc định sử dụng trộn):

$ git pull tên\_nhánh\_máy\_chủ tên\_nhánh\_cục\_bộ

# PHẦN 2: Xây dựng form nhập liệu bằng Excel

## Đặt vấn đề và định hướng giải pháp

### Đặt vấn đề

Các dạng chương trình dạng form nhập liệu, tương tác với người dùng đều có thể lập trình được trên các kiểu giao diện đồ họa như winform, java swing,…. Điểm chung của các giao diện này là dễ tùy biến, lập trình viên có thể thiết kế mọi thứ trên giao diện này. Tuy nhiên có một hạn chế đó là do tính tùy biến nên các giao diện người dùng này không có tính nhất quán và gây khó khăn cho người sử dụng. Đa số đều phải hướng dẫn chi tiết các thông tin trên các form này cho người sử dụng.

Một giải pháp đưa ra đó là sử dụng MS Office để xây dựng form nhập liệu. MS Office là ứng dụng rất phổ biến với người sử dụng nên người sử dụng sẽ rất dễ dàng tương tác với giao diện của ứng dụng. Không những vậy, MS Office còn hỗ trợ bộ công cụ tiện lợi có sẵn cho người dùng và các công cụ riêng cho lập trình viên nên việc lập trình với MS Office không quá khó khăn. Ngoài việc tương tác với giao diện, người dùng còn có thể chỉnh sửa lại giao diện trong phạm vi cho phép của lập trình viên. Vì vậy sử dụng MS Office để xây dựng form nhập liệu mang lại nhiều lợi ích cho cả người sử dụng và lập trình viên. Trong đồ án này sẽ sử dụng Excel để xây dựng form nhập liệu và kết nối đến hệ quản trị cơ sở dữ liệu SQL Server.

### Excel và các công cụ sử dụng

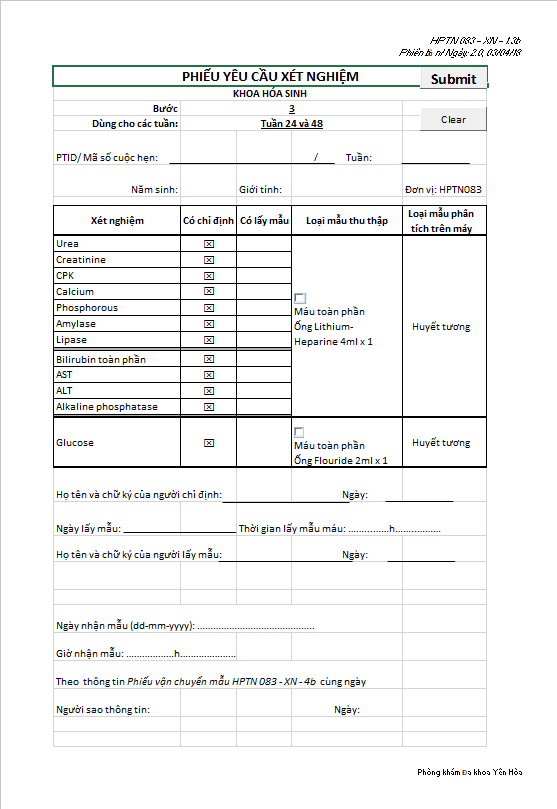
Excel là một chương trình xử lý bảng tính nằm trong bộ Microsoft Office của hãng Microsoft. Các thông tin được trình bày và xử lý dưới dạng bảng. Chính vì các thông tin được hiển thị dưới dạng bảng tính nên việc xây dựng một giao diện trên Excel cũng có dạng bảng tính và thực hiện việc này khá dễ dàng. Ngoài các công cụ cơ bản được cung cấp khi làm việc với Excel như căn chỉnh font chữ lề chữ, các ô trong bảng tính (gọi là các *cells*),…. Excel còn cung cấp một bộ công cụ khổng lồ dành cho lập trình viên nằm trong tab Developer.

Trong tab Developer, Microsoft tích hợp vào bộ Office công cụ Visual Basic for Application (VBA), lập trình viên sử dụng ngôn ngữ lập trình Visual Basic để lập trình trong Excel. Các hàm – api có sẵn trong Excel sẽ giúp cho việc tương tác giữa VBA với bảng tính được thuận tiện hơn. Bằng VBA, lập trình viên có thể tạo các control như button, textbox, listbox,… ngay trên chính giao diện bảng tính và bắt các sự kiện khi người dùng tương tác với các control này. Ngoài ra, VBA còn hỗ trợ thư viện để kết nối với các hệ quản trị CSDL như SQL Server, My SQL,… và thực hiện các câu truy vấn, chỉnh sửa dữ liệu trên CSDL.

Để kết nối đến CSDL, ngoài cách sử dụng VBA ở trên, Excel cung cấp công cụ để thực hiện kết nối và viết các câu truy vấn đến CSDL như: lấy các dữ liệu từ bảng trong CSDL và đưa vào một sheet, thiết lập tham số tự động cập nhật các bảng này sau một khoảng thời gian nhất định hay khi mở file excel.

Một công cụ hữu ích khác được sử dụng trong đồ án là Data validation – kiểm tra tính hợp lệ của dữ liệu. Dữ liệu ở các cells mà người dùng nhập vào là ngẫu nhiên, cần phải được kiểm tra tính hợp lệ của dữ liệu trước khi xử lý hay đẩy vào CSDL, việc làm này là rất quan trọng trong thiết kế form nhập liệu. Thay vì sử dụng VBA để kiểm tra dữ liệu trên cells, việc làm này tốn nhiều thời gian vì VBA không có hàm có sẵn, Excel cung cấp công cụ Data Validation để thực hiện việc đó. Excel sẽ tự động kiểm tra cell mỗi khi người dùng nhập xong dữ liệu cho cell đó và báo lỗi ngay nếu như dữ liệu trong cell không đúng định dạng và điều kiện đã thiết lập trước.

Trong đồ án sẽ sử dụng file Excel có nội dung là một phiếu xét nghiệm như sau:



*Hình 12: File xét nghiệm*

## Phân tích và thiết kế các chức năng của hệ thống

### Các chức năng

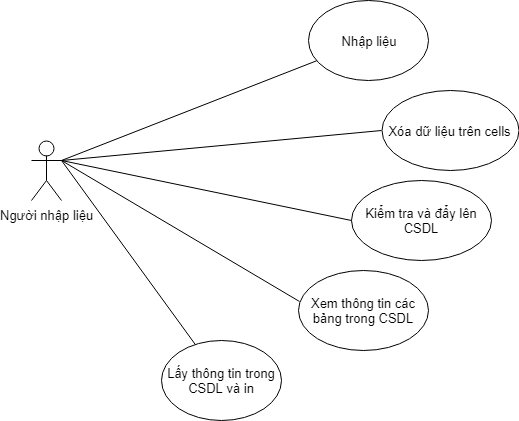
Các chức năng chính của hệ thống nhập liệu phiếu xét nghiệm bao gồm:

* Cho phép người dùng nhập dữ liệu vào các cells đã được chỉ định trước, tránh việc người dùng nhập sai vị trí. Đồng thời khóa các cells mang nội dung của phiếu xét nghiệm, người dùng sẽ không thể sửa nội dung các cells này. Sau đó kiểm tra tính hợp lệ của các ô mà người dùng nhập vào rồi đưa dữ liệu đã nhập vào CSDL.
* Lấy dữ liệu từ các bảng trong CSDL và hiển thị ra các sheet History tương ứng bao gồm: các sheet về các bảng của CSDL, sheet lưu thông tin về số phiếu chỉ định xét nghiệm của từng loại mẫu.
* Đưa thông tin từ CSDL vào form nhập liệu: tiến hành tìm kiếm theo mã số cuộc hẹn trong CSDL do người dùng nhập vào và truy xuất dữ liệu điền thông tin vào form. Người dùng có thể in phiếu xét nghiệm này bằng cách in như thông thường trong Excel.



Hình : Biểu đồ phân rã chức năng

### Use case



Hình : Biểu đồ usecase

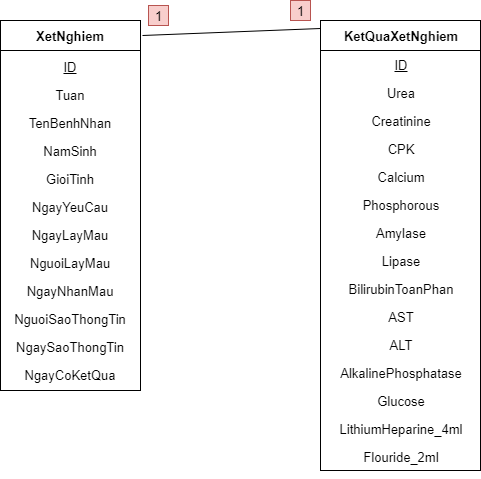
Đặc tả use case nhập liệu phiếu xét nghiệm:

* Nhập liệu:
* Người dùng nhập dữ liệu vào các cells tương ứng.
* Excel hiện thông báo lỗi nếu dữ liệu nhập vào không hợp lệ với cells đó.
* Kiểm tra và đẩy lên CSDL:
* Người dùng nhấn nút Submit.
* Hệ thống kiểm tra các cells đã đầy đủ chưa, kiểm tra tính hợp lệ của các tên.
* Nếu xảy ra lỗi, hiển thị thông tin lỗi và kết thúc.
* Nếu không xảy ra lỗi thì thực hiện kết nối đến CSDL và đẩy thông tin lên CSDL rồi thông báo thành công.
* Xóa dữ liệu trên các cells:
* Người dùng nhấn nút Clear.
* Hệ thống clear toàn bộ nội dung của các cells nhập liệu để chuẩn bị cho lần nhập liệu tiếp theo.
* Xem thông tin các bảng trong CSDL:
* Người dùng chuyển qua từng sheet xem thông tin các bảng.
* Người dùng nhấn nút Refresh All trong tab Data.
* Hệ thống cập nhật lại dữ liệu trên các sheet từ CSDL. Hệ thống được cài đặt tự động cập nhật sau vài phút.
* Lấy thông tin trong CSDL và in:
* Người dùng điền mã số cuộc hẹn vào ô tương ứng rồi nhấn nút Find and Show.
* Hệ thống tìm kiếm trong CSDL và hiện thị thông tin lên form nhập liệu, nếu không có hoặc lỗi nhập sai thì thông báo.
* Người dùng có thể in form nhập liệu như cách in file Excel thông thường (các button được cài đặt không in).

### Xây dựng cơ sở dữ liệu

#### **Thiết kế cơ sở dữ liệu**

Cơ sở dữ liệu xét nghiệm dùng để lưu trữ thông tin và truy vấn thông tin đối với form nhập liệu trên. Các trường trong CSDL tương ứng với thông tin được nhập vào từ bảng trong Excel. Sơ đồ quan hệ thực thể:



Hình : Sơ đồ thực thể quan hệ

Bảng : Bảng XetNghiem trong CSDL

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Điều kiện** | **Ghi chú** |
| ID | Int | Khóa chính, ID > 0 | Mã số cuộc hẹn |
| Tuan | Int | NOT NULL, (0, 53] | Tuần |
| TenBenhNhan | Nvarchar(100) | NOT NULL | Tên người chỉ định |
| NamSinh | Int | NOT NULL | Năm sinh người chỉ định |
| GioiTinh | Nvarchar(5) | NOT NULL | Giới tính người chỉ định |
| NgayYeuCau | Date | NOT NULL | Ngày yêu cầu |
| NgayLayMau | Datetime | NOT NULL | Ngày lấy mẫu |
| NguoiLayMau | Nvarchar(100) | NOT NULL | Người lấy mẫu |
| NgayNhanMau | Datetime | NOT NULL | Ngày nhận mẫu |
| NguoiSaoThongTin | Nvarchar(100) | NOT NULL | Người sao thông tin |
| NgaySaoThongTin | Date | NOT NULL | Ngày sao thông tin |
| NgayCoKetQua | Date | NOT NULL | Ngày có kết quả |

Bảng : Bảng KetQuaXetNghiem trong CSDL

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Điều kiện** | **Ghi chú** |
| ID | Int | Foreign key | Tham chiếu đến trường ID trong bảng XetNghiem |
| Urea | Decimal(5, 2) | Số thực dương | Nồng độ Urea |
| Creatinine | Decimal(5, 2) | Số thực dương | Nồng độ Creatinine |
| CPK | Decimal(5, 2) | Số thực dương | Nồng độ CPK |
| Calcium | Decimal(5, 2) | Số thực dương | Nồng độ Calcium |
| Phosphorous | Decimal(5, 2) | Số thực dương | Nồng độ Phosphorous |
| Amylase | Decimal(5, 2) | Số thực dương | Nồng độ Amylase |
| Lipase | Decimal(5, 2) | Số thực dương | Nồng độ Lipase |
| BilirubinToanPhan | Decimal(5, 2) | Số thực dương | Nồng độ Bilirubin toàn phần |
| AST | Decimal(5, 2) | Số thực dương | Nồng độ AST |
| ALT | Decimal(5, 2) | Số thực dương | Nồng độ ALT |
| AlkalinePhosphatase | Decimal(5, 2) | Số thực dương | Nồng độ Alkaline Phosphatase |
| Glucose | Decimal(5, 2) | Số thực dương | Nồng độ Glucose |
| LithiumHeparine\_4ml | BIT | NOT NULL | Check loại mẫu thu nhập |
| Flouride\_2ml | BIT | NOT NULL | Check loại mẫu thu nhập |

#### 

#### **Kết nối đến SQL Server**

VBA cung cấp đối tượng ADODB.Connection trong thư viện Microsoft ActiveX Data Objects 2.8 để khởi tạo kết nối đến CSDL thông qua ODBC. Open Database Connectivity - ODBC là một phương pháp truy xuất CSDL chuẩn được phát triển bởi Microsoft. ODBC cho phép các ứng dụng truy cập dữ liệu trong các hệ quản trị CSDL bằng cách sử dụng SQL làm tiêu chuẩn để truy cập dữ liệu. ODBC cung cấp cho các ứng dụng khả năng truy xuất dữ liệu bất kì mà không phải quan tâm đến việc hiện tại dữ liệu đang được quản lý bởi hệ quản trị CSDL nào. ODBC làm được việc này bằng cách chèn một lớp trung gian vào giữa trình ứng dụng và hệ quản trị CSDL. Lớp trung gian đó được gọi là một Database Driver (trình điểu khiển CSDL). Mục đích của lớp này là chuyển đổi những câu truy vấn của trình ứng dụng thành những lệnh mà hệ quản trị CSDL hiểu. Để làm được việc này thì cả trình ứng dụng và hệ quản trị CSDL phải hiểu biết ODBC. SQL Server được tích hợp sẵn driver trong ODBC.

Để kết nối đến SQL Server thông qua ODBC thì connection string chuẩn có dạng như sau:

Driver={SQL Server};Server=myServerAddress;Database=myDatabase;Uid=Username;Pwd=myPassword;

Và đây là đoạn mã Visual Basic kết nối đến SQL Server thông qua ODBC.

Dim Cn As ADODB.Connection

Set Cn = New ADODB.Connection

Cn.Open “điền connection string ở đây”

‘ Thao tác truy vấn dữ liệu

‘ ..

‘ ..

Cn.Close

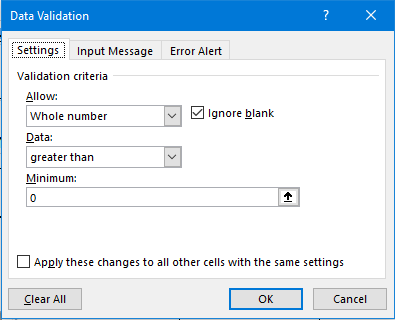
### 

### Thiết kế các chức năng

#### **Form nhập liệu**

File Excel đã được người dùng thiết kế sẵn giao diện, việc cần làm là phải chỉ định cells nhập dữ liệu và kiểm tra dữ liệu nhập vào. Giải pháp đưa ra là dùng Data Validation, trong công cụ này có các tùy chọn bao gồm: khi chọn một cell thì hiện comment cho cell đó – sử dụng để gợi ý người dùng nhập dữ liệu tại đây; đặt điều kiện về dữ liệu cho cell đó đồng thời hiện thông báo nếu điều kiện sai – sử dụng để hiển thị lỗi khi người dùng nhập không đúng. Ví dụ với ô ID – mã số cuộc hẹn được thiết lập Data Validation như sau:

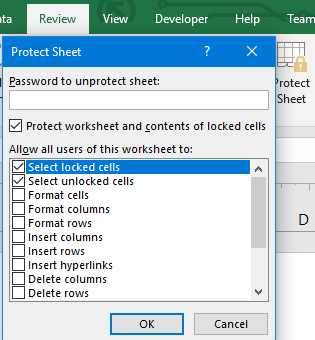
* Tab Settings:
* Allow: kiểu dữ liệu
* Data: loại so sánh
* Tùy từng loại so sánh mà có thêm các thông tin về giá trị
* Tab Input Message:
* Cài đặt để hiện nội dung comment (hiện ngay dưới cell) khi người dùng chọn vào cell này.
* Tab Error Alert:
* Cài đặt để hiện nội dung thông báo khi người dùng nhập vào dữ liệu không thỏa mãn điều kiện đã đặt ra ở tab Settings.



Hình : Data Validation của mã số cuộc hẹn

Tương tự đối với tất cả các cells người dùng nhập vào khác, đối với các ô có dữ liệu là ngày tháng cần phải đặt cả Number Format về kiểu dữ liệu tương ứng để dữ liệu hiện trên các ô đúng dạng, tùy chọn này nằm ở Tab Home.

Ngoài ra, người dùng chỉ được phép truy cập hay chỉnh sửa các ô được cho phép, không được phép chỉnh sửa form nhập liệu, do vậy cần khóa – bảo vệ sheet nhập liệu. Để làm điều này trong Excel có công cụ Protect Sheet trong Tab Review, khóa sheet bằng mật khẩu. Để chỉnh định các cell bị khóa, ta chọn cell tương ứng và click chuột phải vào tùy chọn Format Cells. Tại đây có tất cả các thiết lập cho cell này như: kiểu dữ liệu (number format), font, căn chỉnh chữ,…. Trong tab Protection nếu chọn Locked thì cell sẽ bị khóa và người dùng không thể chỉnh sửa khi tính năng Protect Sheet được bật. Mặc định, Protect Sheet sẽ thiết lập khóa toàn bộ các chức năng liên quan đến chỉnh sửa file Excel như: thêm sửa xóa cột, truy cập format cells, chỉnh sửa data validation,… mà chỉ cho phép sửa nội dung các cells không bị khóa và một số tính năng khác liên quan đến view hay truy xuất vào CSDL.



Hình : Tùy chọn trong Protect Sheet

#### **Kết nối và đưa dữ liệu lên cơ sở dữ liệu**

Chức năng này được thực hiện khi người dùng nhấn vào nút Submit được tạo bằng ActiveX Control trên góc phải của form. Sự kiện khi người dùng nhấn vào nút Submit được bắt và thực hiện bằng mã visual basic trong hàm btnSubmit\_Click(). Hàm này sẽ thực hiện kiểm tra tất cả các cells đã nhập đầy đủ hay chưa và kiểm tra dữ liệu được nhập. Đối với dữ liệu dạng date, datetime, number thì chỉ cần kiểm tra cell này có trống hay không vì nội dung cell đã được kiểm tra bởi Data Validation khi người dùng nhập. Còn dữ liệu dạng tên tiếng việt thì phải kiểm tra lại. Đồng thời cần format lại tên: xóa các khoảng trắng thừa, viết hoa chữ đầu tiên. Các hàm kiểm tra về dữ liệu sẽ nằm trong module *BusinessDataValidation.*

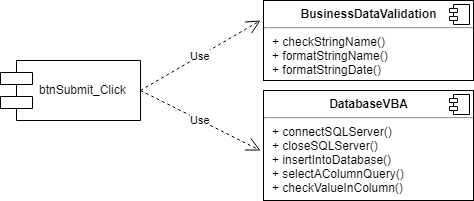
Module BusinessDataValidation:

* Hàm checkStringName: kiểm tra tên tiếng việt – phải là ký tự Unicode và không chứa số hay các dấu đặc biệt.
* Hàm formatStringName: thực hiện format chuỗi tên tiếng việt sau khi kiểm tra là hợp lệ.
* Hàm formatStringDate: thực hiện nối ngày/tháng/năm với giờ/phút/giây của cùng một đối tượng (nếu có).

Ngoài ra, cần phải truy xuất CSDL để kiểm tra ID đã tồn tại trong CSDL hay chưa. Sau khi kiểm tra hoàn tất, nếu không có lỗi thì thực hiện kết nối đến SQL Server và thực hiện các câu truy vấn Insert để đưa dữ liệu vào CSDL. Các hàm trong Module DatabaseVBA sẽ thực hiện chức năng này.

Module DatabaseVBA:

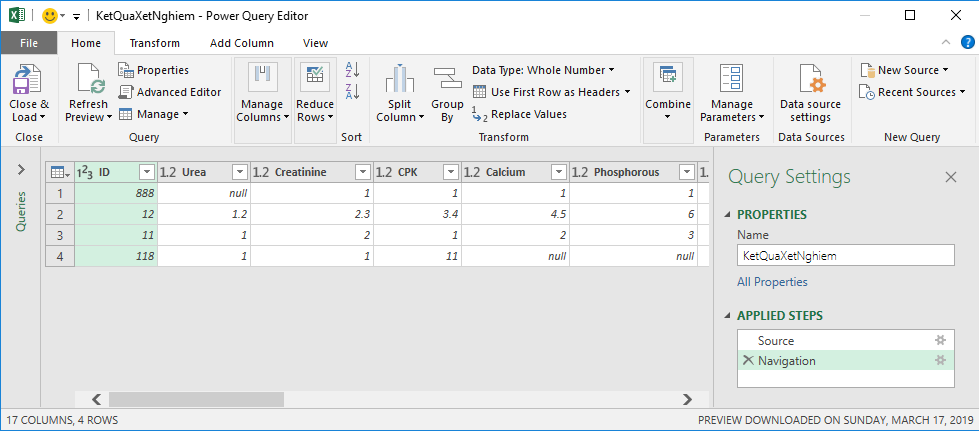
* connectSQLServer(): kết nối đến SQL Server, tham số truyền vào là tên server, tên CSDL, user và pass, trả về một kiểu ADODB.Connection làm tham số đầu vào cho các hàm khác truy xuất tới CSDL.
* closeSQLServer(): đóng kết nối, tham số đầu vào là một ADODB.Connection.
* insertIntoDatabase(): thực hiện câu lệnh insert, tham số đầu vào là câu truy vấn.
* selectAColumnQuery(): câu lệnh select một cột trong CSDL, tham số đầu vào là tên bảng và tên cột.
* checkValueInColumn(): kiểm tra dữ liệu trong một cột, sử dụng để kiểm tra ID đã tồn tại chưa. Tham số đầu vào là tên bảng, tên cột và giá trị.



*Hình 18: Biểu đồ triển khai btnSubmit\_Click*

#### **Lấy dữ liệu từ cơ sở dữ liệu hiển thị lên sheet History**

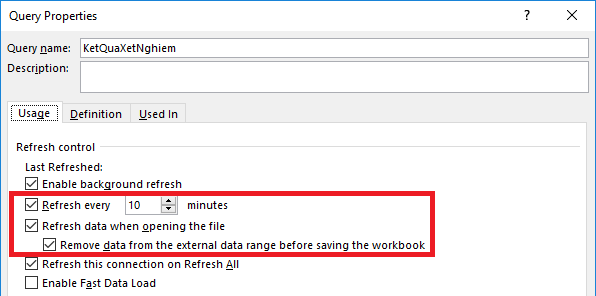
Excel hỗ trợ công cụ kết nối với CSDL trong tab Data. Để thiết lập kết nối tới SQL Server, chọn tab Data > Get Data > From Database > From SQL Server Database và điền thông tin tên server, tên database, thông tin đăng nhập. Nếu kết nối thành công, thông tin các bảng trong CSDL sẽ được hiện lên và chọn một bảng hoặc nhiều bảng để tiến hành câu truy vấn, Power Query Editor sẽ hiện lên.



Hình : Power Query Editor

Power Query Editor sẽ hỗ trợ tạo câu truy vấn theo kiểu giao diện trên, để tạo câu truy vấn, thực hiện chọn các yêu cầu tương ứng trên thanh công cụ như: Group By, Manage Colums,…. Sau khi hoàn thành và nhấn Close & Load, Excel sẽ tự động thực hiện câu truy vấn và lấy kết quả hiển thị lên một bảng ở một sheet mới.

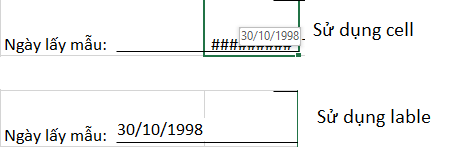
Trong tab Data, công cụ Refresh All sẽ cập nhật toàn bộ những câu truy vấn được tạo theo cách trên. Ta có thể thiết lập để Excel tự động cập nhật trong khoảng thời gian nhất định hoặc khi mở file Excel này lên bằng cách truy cập Queries & Connections và tìm câu truy vấn tương ứng, sau đó click chuột phải chọn Query Properties.

*Hình 20: Thiết lập Query Properties*

#### **Lấy dữ liệu từ cơ sở dữ liệu hiển thị lên form**

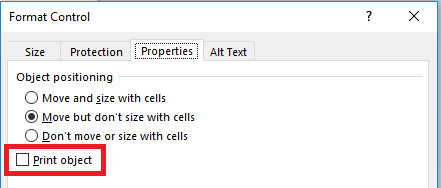
Mục đích của tính năng này là giúp người dùng có thể dễ dàng kiểm tra thông tin của từng phiếu xét nghiệm hoặc có thể in lại phiếu xét nghiệm có trong CSDL. Trong tính năng này, việc kết nối và lấy dữ liệu từ CSDL được thực hiện bằng VBA. Các thao tác kết nối và viết truy vấn giống trong phần lưu dữ liệu. Tính năng được thực hiện khi người dùng click vào nút Find & Show, hàm callback btnFindShow\_Click() sẽ được gọi và thực thi. Dữ liệu sẽ được đẩy lên từng cells tương ứng trong form nhập liệu.

Tuy nhiên việc đẩy dữ liệu vào cell như khi nhập sẽ dẫn đến mất cân xứng giữa tiêu đề và dữ liệu, một số cell có kích thước nhỏ sẽ không hiển thị hết dữ liệu. Giải pháp là sử dụng Label thay thế cho từng cell này. Khi người dùng nhập dữ liệu thì các Labels sẽ ẩn đi bằng cách set thuộc tính *“visible = false”*, khi người dùng thực hiện lấy dữ liệu vào form thì sẽ hiện Label lên.



Hình : Sử dụng Label để hiển thị

Người dùng có thể nhấn Ctrl + P và thực hiện in như bình thường. Các nút tương tác với người dùng sẽ bị ẩn đi. Để thiết lập ẩn cho các nút, click chuột phải vào nút tương ứng và chọn Format Control, chuyển sang tab Properties và bỏ chọn “Print object”.

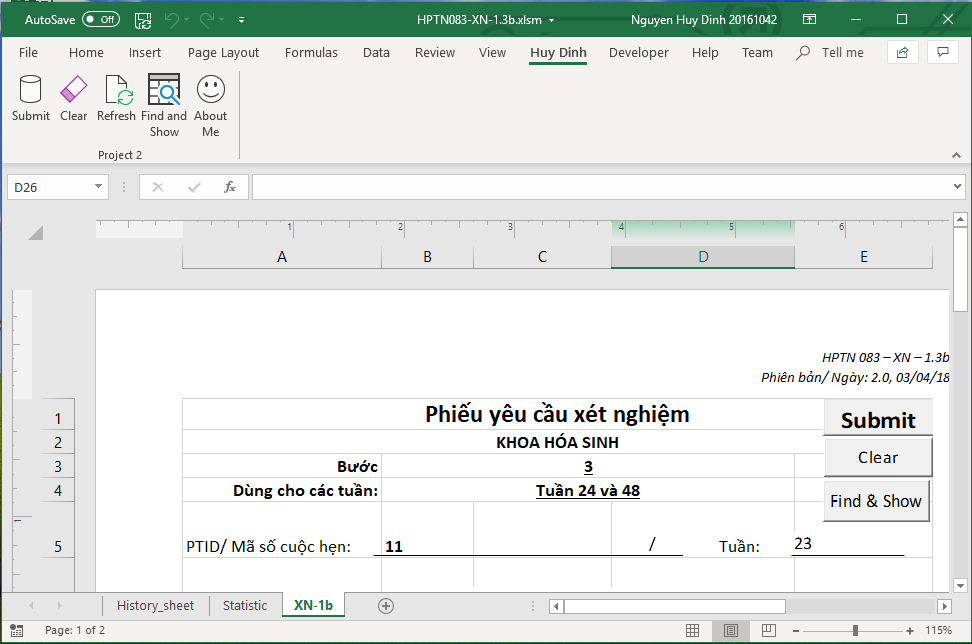


Hình : Thiết lập in các ActiveX Controls.

## KẾT LUẬN

Kết quả đạt được:

* Hoàn thành yêu cầu của form nhập liệu bằng Excel với các tính năng cơ bản: lưu dữ liệu vào CSDL, lấy ra dữ liệu từ CSDL.
* Ngoài các nút được bố trí trên form nhập liệu và các tab có sẵn, có thêm các nút có chức năng tương tự trong tab tự thiết kế trên thanh Ribbon bằng công cụ Custom UI Editor For MS.



Hình : Giao diện tương tác với người dùng của chương trình

Hạn chế của chương trình:

* Giao diện nhập liệu chưa được trực quan, phụ thuộc vào file Excel ban đầu của người dùng.
* Gợi ý ô nhập liệu chưa thực sự hiệu quả, các ô nhập liệu có kích thước hạn chế bởi kích thước của cell.

Các hướng phát triển:

* Thêm các tính năng cho người dùng như: tìm kiếm thông tin, xóa thông tin trong CSDL.
* Xây dựng phần mềm quản lý nhập liệu trên Excel.

# TÀI LIỆU THAM KHẢO

**\* *Danh mục sách:***

**\* *Danh mục tạp chí:***

**\* *Danh mục hội thảo:***

**\* *Danh mục internet:***

1. Qui định về đồ án tốt nghiệp của Viện CNTT-TT  
   <https://soict.hust.edu.vn/index.php/2017/03/16/quy-dinh-ve-do-an-tot-nghiep/>
2. Git Book, [https://git-scm.com/book/vi/](https://git-scm.com/book/vi/v1/B%E1%BA%AFt-%C4%90%E1%BA%A7u-C%C6%A1-B%E1%BA%A3n-v%E1%BB%81-Git)
3. Stackoverflow, <https://stackoverflow.com/>
4. VBA in Excel, <https://docs.microsoft.com/en-us/office/vba/api/overview/excel>