

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
Viện Toán ứng dụng và Tin học
---&BOOK&---



Báo Cáo Bài Tập Lớn

Đề 2: Thực hiện viết chương trình quản lý dự án.

*Học phần: Kỹ thuật lập trình
Giáo viên hướng dẫn: Nguyễn Thị Thanh Huyền
Sinh viên thực hiện: Đinh Đức Đạt
Mã số sinh viên: 20216809
Lớp: MI1- 02*



MỤC LỤC

MỤC LỤC.....	1
Phần 1. Giới thiệu, phân tích yêu cầu đối với chức năng của phần mềm.....	2
1.1 Giới thiệu phần mềm.	
1.1.1. Các yêu cầu đối với chương trình.	
1.1.2. Ý nghĩa thực tiễn của chương trình	
1.2 Phân tích các chức năng của chương trình.	
1.2.1. Các chức năng chính của chương trình.	
1.2.2. Sơ đồ các chức năng của chương trình.	
1.3 Các thư viện được sử dụng trong chương trình.	
1.3.1. Giới thiệu về các thư viện được sử dụng trong chương trình.	
1.3.2. Cài đặt thư viện vào hệ thống.	
Phần 2. Phương án hoàn thành chương trình.....	
2.1 Ý tưởng hoàn thành chương trình.	
2.2 Sơ đồ khái của chương trình.	
2.2.1. Sơ đồ khái của hệ thống.	
2.2.3. Quá trình thực hiện chương trình.	
2.2.4.	
2.3 Các thuật toán và cấu trúc dữ liệu sử dụng trong chương trình.	
2.3.1. Cấu trúc dữ liệu trong đọc và xử lý dữ liệu.	
2.3.2. Cấu trúc dữ liệu được sử dụng trong xử lý MENU người dùng.	
2.4 Phương án phân chia các chức năng của của chương trình.	
2.4.1 Phân chia các chức năng của chương trình.	
2.4.2 Phương án xử lý các vấn đề của chương trình.	
2.4.3 Phương án xử lý và hoàn thiện chương trình.	
Phần 3. Thực hiện chương trình.....	
3.1 Phân tích cấu trúc chương trình.	
3.1.1 Cấu trúc chương trình.	
3.2 Các cấu trúc dữ liệu của chương trình sử dụng.	
3.3 Các giải thuật mà chương trình sử dụng.	
3.4 Mã nguồn của chương trình.	
Phần 4. Kiểm thử chương trình.....	
4.1 Khả năng hoạt động của chương trình.	
4.1.1 Đánh giá mức ổn định của chương trình.	
4.1.2 Đánh giá khả năng hoạt động đối với người dùng.	
4.1.3 Giao diện người dùng.	
4.2 Kiểm thử các chức năng của hệ thống.	
4.2.1 Chức năng quản lý.	
4.2.2 Chức năng lưu trữ.	
4.2.3 Chức năng tìm kiếm.	
4.3 Đánh giá mức độ hoàn thiện của chương trình.	

Phần 1. Giới thiệu, phân tích yêu cầu và chức năng đối với phần mềm.

1.1 Giới thiệu phần mềm.

1.1.1 Các yêu cầu đối với phần mềm.

Phần mềm được viết ra với mục đích quản lý nhân viên trong các dự án lớn với sự tham gia của rất nhiều nhân viên. Việc quản lý toàn bộ nhân viên này là tương đối phức tạp. Thế nên trong quá trình hoàn thiện chương trình các yêu cầu đối với hệ thống là tương đối cao.

Thứ nhất, dữ liệu được lưu trong hệ thống phải được độc lập và không được xung đột với nhau. Điều này là cần thiết bởi dữ liệu được lưu trong hệ thống là dữ liệu quan trọng mà nếu xảy ra sai sót thì để lại hậu quả to lớn đối với công ty. Thế nên các dữ liệu được nhập vào phải đảm bảo không bị sai sót cũng như không để trùng lặp dữ liệu, không có bản ghi nào xung đột với nhau.

Thứ hai, Giao diện người dùng chương trình phải thân thiện với người sử dụng. Trong thực tế người sử dụng phần mềm này không phải là các lập trình viên hay là nhà phát triển phần mềm mà hoàn toàn là các nhân viên kế toán. Các nhân viên Kế toán họ không được đào tạo sâu về nghiệp vụ IT thế nên nếu chương trình được viết với giao diện dòng lệnh thì sẽ rất khó đảm bảo cho các nhân viên kế toán có thể tiếp cận với phần mềm một cách dễ dàng được. Vì họ sẽ mất rất nhiều thời gian để làm quen và học tập các câu lệnh đối với hệ thống. Việc tạo ra một chương trình có giao diện người dùng thân thiện sẽ là tiền đề để giảm thiểu chi phí đào tạo sử dụng phần mềm. Từ đó phần mềm sẽ được ứng dụng nhiều hơn trong thực tiễn, tăng doanh thu đối với các công ty sử dụng phần mềm này.

Thứ ba, đảm bảo tính ổn định đối với lượng dữ liệu lớn và thời gian sử dụng kéo dài. Trong thực tiễn các phần mềm sẽ được sử dụng trong một thời gian rất dài nếu chương trình được viết một cách không chỉnh chu sẽ gây ra các hiện tượng như rò rỉ bộ nhớ (Đối với các ngôn ngữ bậc trung) hoặc mất mát dữ liệu (Đối với lượng dữ liệu tương đối lớn)... gây ảnh hưởng tiêu cực tới quá trình hoàn thành dự án đề ra.

Thứ tư, Phần mềm được tạo ra phải hoạt động mượt mà, tối ưu thời gian chạy của các chức năng.

1.1.2 Ý nghĩa thực tiễn của phần mềm.

Trong thực tế quá trình xây dựng và quản lý các dự án không thể tránh khỏi nhiều khó khăn trong việc lưu trữ và quản lý dự án. Việc tạo ra một phần mềm để quản lý các nhân viên trong dự án là hoàn toàn cấp thiết. Việc tạo ra những chương trình giúp quản lý các nhân viên trong dự án giúp giảm thiểu chi phí và tránh các sai sót đối với các công việc có lượng dữ liệu lớn.

Phần mềm được tạo ra nhằm mục đích quản lý nhân viên trong các dự án của các công ty, giúp cho công tác quản lý nhân sự trong các dự án được tiến hành một cách đồng bộ. Giảm thiểu lãng phí nhân sự cho công tác quản lý, hạn chế các sai sót không đáng có.

Trong quá trình thực hiện phần mềm, vì trình độ còn hạn chế và thời gian hạn hẹp không thể tránh khỏi các lỗi nhỏ trong phần mềm.

1.2 Phân tích các chức năng của phần mềm.

1.2.1 Các chức năng giao diện người dùng.

Các chức năng giao diện người dùng cung cấp cho người dùng các phương thức nhập, xuất, điều khiển từ chuột hoặc bàn phím thay vì sử dụng giao diện dòng lệnh gây khó khăn cho người mới tiếp cận phần mềm. Việc tạo ra giao diện người dùng giúp cho người dùng được thoải mái hơn trong quá trình thao tác đối với hệ thống phần mềm.

Đây là phần mềm được viết thủ công bằng tay sử dụng thư viện đồ họa SDL.h là một thư viện xử lý đồ họa trên C++ cung cấp các chức năng, các phương thức các hàm đồ họa cho phép người sử dụng có thể vẽ cá đối tượng 2D lên màn hình máy tính. Thư viện này đơn thuần chỉ là thư viện đồ họa, nên không hỗ trợ nhập xuất từ bàn phím nên ta hoàn toàn phải tạo ra các hàm nhập xuất từ bàn phím một cách thủ công.

Đối với chức năng điều khiển của người dùng, Thì người dùng sẽ điều khiển chương trình thông qua chuột và bàn phím. Các nút được sử dụng là chuột trái, chuột phải. Khi ta nhấn chuột vào các phím chức năng thì giao diện xử lý sẽ được thực hiện.

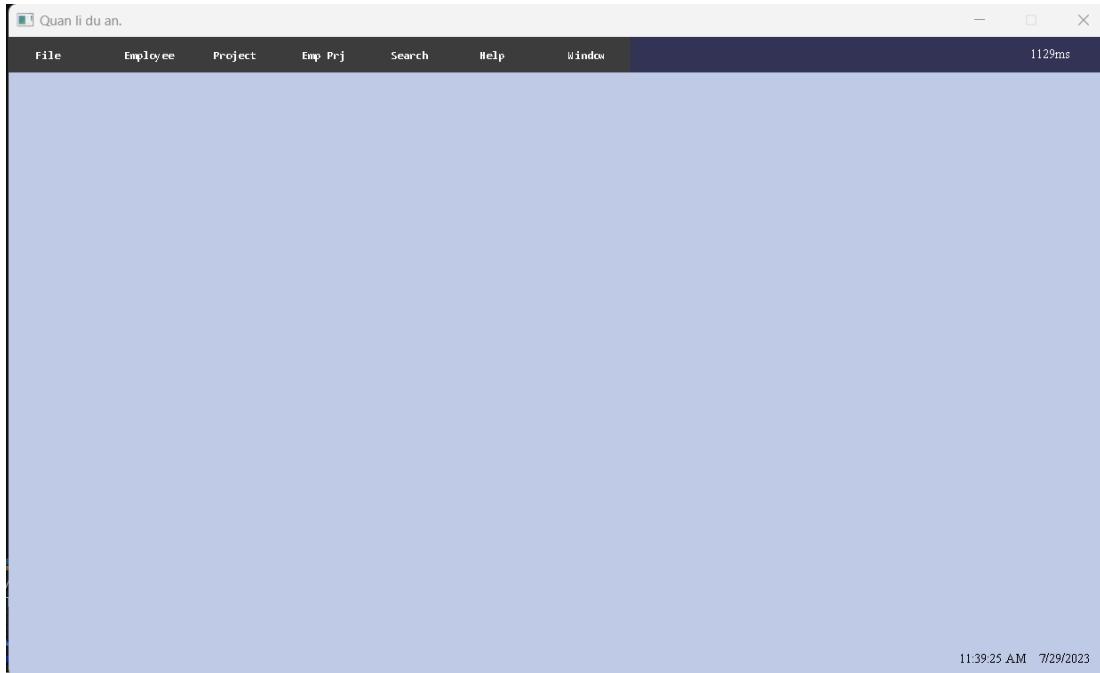


Figure 1. hình ảnh chương trình sau khi khởi tạo.

Có thể thấy trong chương trình có sử dụng một thanh Menu gồm nhiều tùy chọn người dùng. Các chức năng này sẽ được phân tích ở mục sau. Trong các tùy chọn này lại có thêm các tùy chọn cấp hai sẽ được hiện ra khi người dùng nhấn chuột trái vào trong menu cấp một.

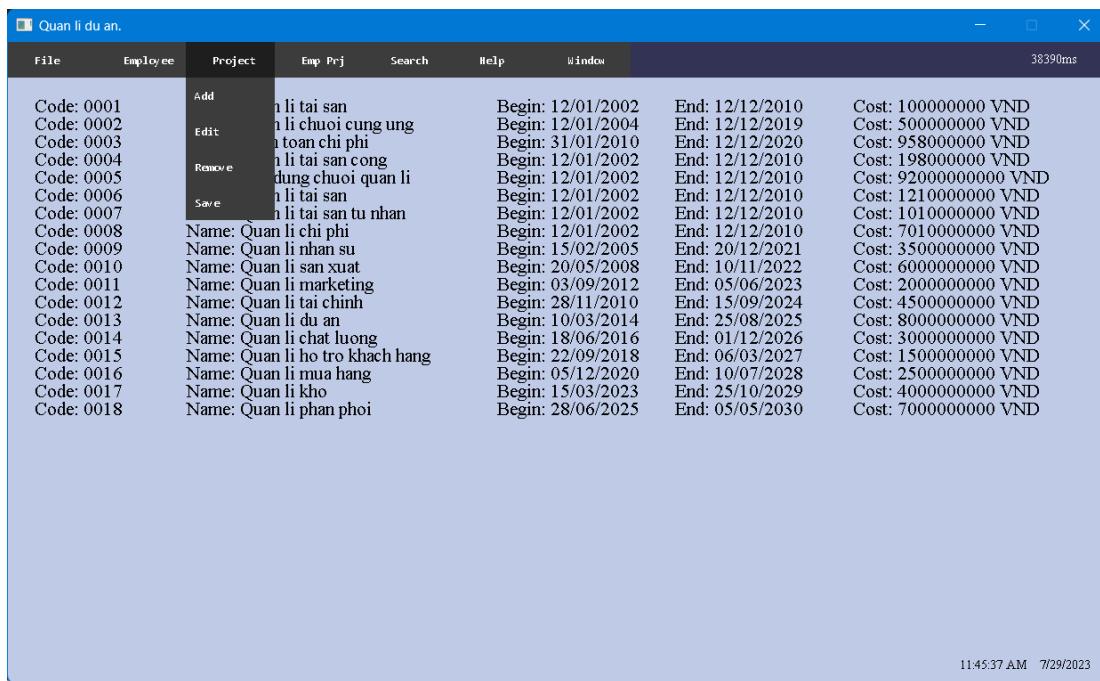


Figure 2 Menu cấp hai.

Sau khi nhấn vào cá tùy chọn của thanh Menu thì ta thấy hiện thêm các chức năng cấp hai. Đối với chương trình này giao diện menu cấp hai có thể là cá tùy chọn.

Add: Thêm một bản ghi mới vào danh sách các bản ghi.

Edit: Chính một bản ghi đã có trong danh sách các bản ghi đã có.

Remove: Xóa một bản ghi đã tồn tại trong danh sách.

Save: lưu toàn bộ danh sách các bản ghi.

1.2.2 Các chức năng nhập xuất và chỉnh sửa dữ liệu.

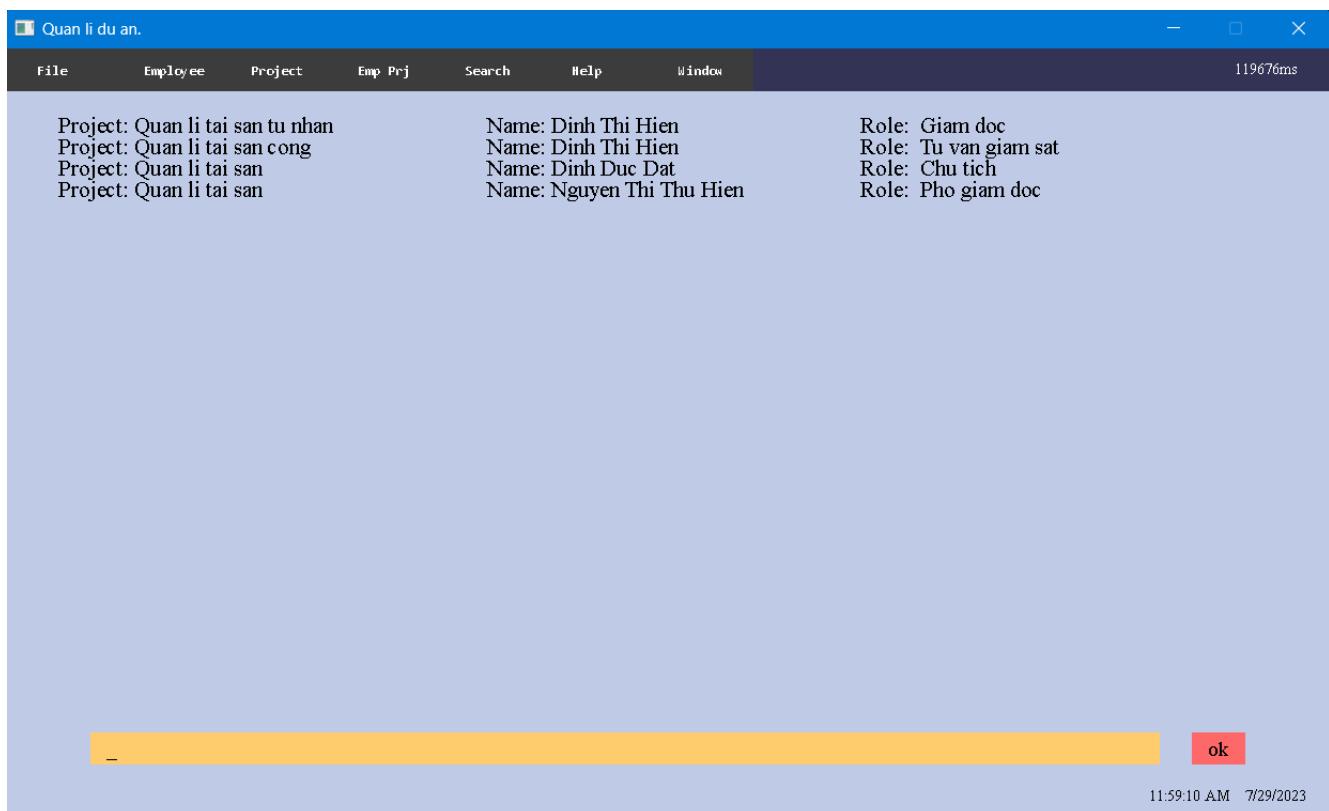


Figure 3. Giao diện nhập dữ liệu.

Sau khi hoàn thành nhập xuất người dùng nhấn chuột trái vào nút “ok” thì dữ liệu sẽ được lưu vào bộ nhớ tạm và được hiển thị lên màn hình. Tiếp theo người dùng có thể tiếp tục thêm, xóa, sửa dữ liệu cho đến khi hoàn thành công việc thì người dùng có thể nhấn vào ô menu cấp hai là “Save” để lưu toàn bộ công việc đã hoàn thành. Nếu người dùng không nhấn chuột vào ô chức năng này mà thoát chương trình thì hệ thống coi như người dùng không muốn thay đổi dữ liệu và tiến hành thoát.

Vì thời gian hạn hẹp và khối lượng công việc là tương đối nhiều nên em không thể thêm các chức năng như tự động sao lưu, hoàn tác,..v.v. mong Cô thông cảm!

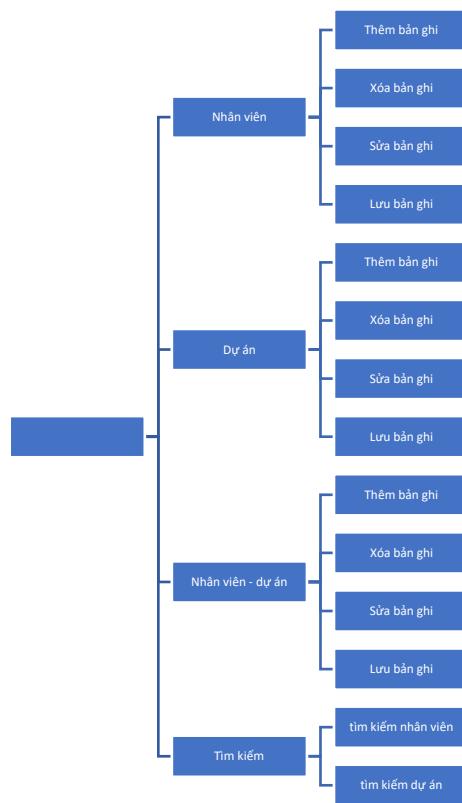
Đối với các danh sách bản ghi như Employee, Project, Employee-Project các thao tác là hoàn toàn tương tự nhau.

1.2.3 Các thông số thể hiện trên màn hình.

Ở góc phải trên cùng của màn hình là thời gian chạy chương trình tính bằng ms.

Ở góc phải bên dưới là thời gian thực hiện chương trình bao gồm các trường: Ngày, tháng, năm, giờ, phút, giây.

1.2.3 Sơ đồ các chức năng của chương trình.



1.3 Các thư viện được sử dụng trong chương trình.

1.3.1. Giới thiệu về các thư viện được sử dụng trong chương trình.

Trong chương trình này sử dụng khá nhiều thư viện các thư viện được sử dụng với mục đích như: hiển thị giao diện người dùng, hiển thị thời gian hệ thống, lưu trữ và ghi file. Danh sách các thư viện được sử dụng trong chương trình gồm:

- Thư viện SDL: thư viện đồ họa có chức năng hiển thị, vẽ giao diện người dùng.



Figure 4. Thư viện SDL

- Thư viện <ctime>: thư viện cho phép lấy thời gian thực của hệ thống.
- Thư viện <fstream> : thư viện cho phép đọc và ghi file.

1.3.2. Tiến hành cài đặt thư viện vào hệ thống.

Đầu tiên chúng ta thực hiện tải thư viện SDL và tiến hành giải nén tập tin vừa được tải về.

Name	Date modified	Type	Size
cmake	6/4/2022 2:53 AM	File folder	
docs	7/2/2023 12:06 AM	File folder	
include	7/2/2023 12:06 AM	File folder	
lib	4/29/2022 11:41 PM	File folder	
BUGS.txt	7/2/2023 12:06 AM	Text Document	1 KB
COPYING.txt	12/22/2020 12:48 AM	Text Document	1 KB
README.txt	12/22/2020 12:48 AM	Text Document	1 KB
README-SDL.txt	7/2/2023 12:06 AM	Text Document	1 KB
WhatsNew.txt	7/2/2023 12:06 AM	Text Document	54 KB

Figure 5. Tập tin của thư viện SDL sau khi được giải nén.

Sau khi giải nén tập tin ta tiếp tục cấu hình thư viện SDL vào trình biên dịch. Cụ thể để thực hiện chương trình này em đã sử dụng thư Visual Studio 2022.

Bước 1. Ta thêm đường dẫn các file thư viện vào C/C++ / General/Additional include Directories. Ta điền đường dẫn tới thư mục include vào trường General/Additional include Directories

Bước 2. Tiếp theo ta tiến hành thêm các file .lib trong thư mục Linker/Input/Additional Dependancies/ ta điền các file.lib có sử dụng trong chương trình.

Bước 3. Ta tiến hành thêm thư mục chứa các file .lib vào trường Linker/General/Additional Library Directory. Ta điền đường dẫn tuyệt đối đến thư mục chứa file lib vào trường này

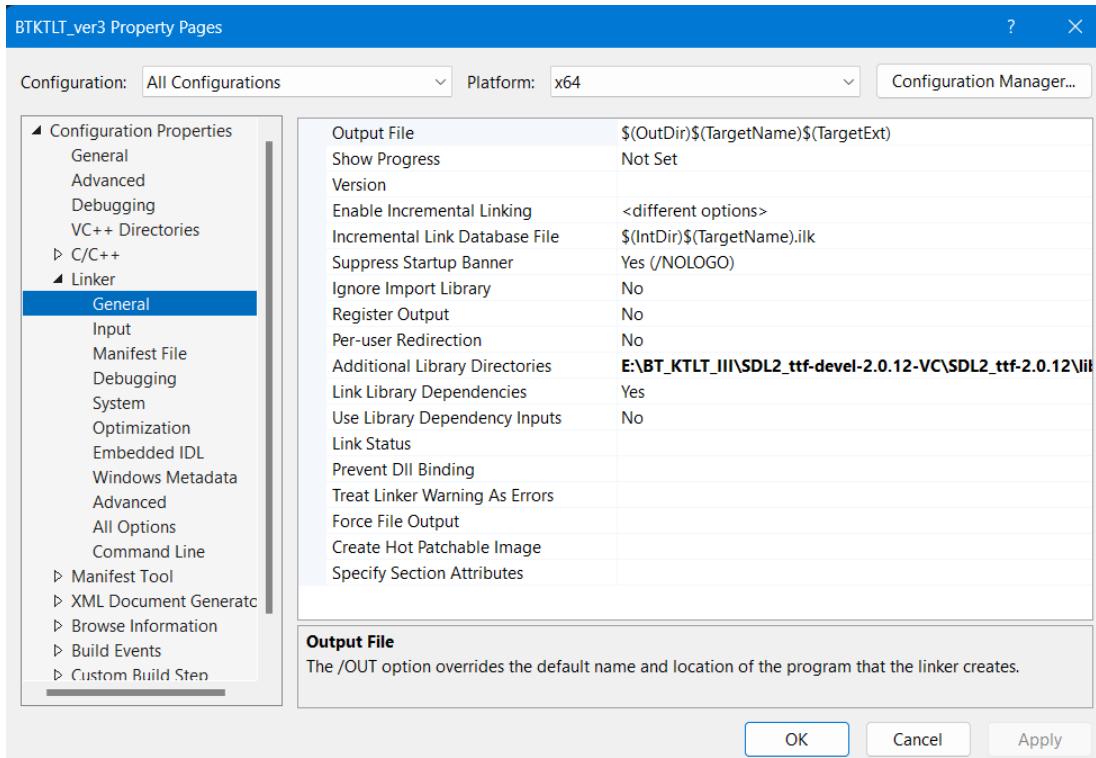


Figure 6. Thêm đường dẫn tới thư mục chứa các file lib

Sau khi cấu hình tiến hành Apply và bắt đầu sử dụng thư viện một cách bình thường.

Phần 2. Phương án hoàn thành chương trình.

2.1 Ý tưởng hoàn thành chương trình.

Nhận thấy đối với một chương trình có giao diện người dùng thì ta có thể tách biệt chương trình thành hai thành phần riêng biệt và có thể hoàn thành cùng lúc đó là phần giao diện sử dụng và phần xử lý dữ liệu. Tuy nhiên trong phần mềm này chỉ do một người thực hiện, nên lúc này phần giao diện người dùng sẽ được tiến hành hoàn thành trước sau đó tiến hành thực hiện phần xử lý thông tin. Vì yêu cầu của đề bài là phải xây dựng một giao diện người dùng một cách thủ công nên phần xây dựng giao diện người dùng này cũng chính là phần phức tạp nhất.

2.1.1 Ý tưởng Xây dựng giao diện người dùng.

Việc xây dựng giao diện người dùng một cách thủ công là phức tạp nên ta sẽ chia nhỏ phần công việc này thành các phần công việc nhỏ hơn nữa để thuận tiện hơn trong công việc.

Thứ nhất, Xây dựng một tập hợp các hàm có chức năng hiển thị các đối tượng được sử dụng trong chương trình. Việc vẽ các đối tượng này bao gồm các hàm vẽ hình vuông, vẽ hình chữ nhật,... Đối với các đối tượng có hiển thị chuỗi ký tự thì ta tiếp tục xây dựng thêm hàm hiển thị ký tự lên màn hình. Các hàm thuộc loại vẽ và hiển thị ký tự ta sẽ gộp lại thành hai tập tin .h riêng biệt.

Thứ hai, Xây dựng một cấu trúc dữ liệu thể hiện các đối tượng trên màn hình giao diện người dùng. Ta có thể phân chia thành nhiều struct nhỏ tuy nhiên đối với bài tập lớn này việc phân chia quá nhiều cấu trúc nhỏ sẽ khiến chương trình bị rối nên ta chỉ chia thành các cấu trúc sau đây:

Cấu trúc H_menu : Thể hiện cho các đối tượng menu items có thể là menu items cấp 1 hoặc là menu items cấp 2. Cấu trúc của cấu trúc này bao gồm các trường dữ liệu : Kích thước của menu items, tọa độ hiển thị của menu items, màu sắc của menu items, trạng thái của menu items (trạng thái của menu items có thể là đang được trỏ, đang được chọn), con trỏ tới menu_item kế tiếp, con trỏ tới menu_item cấp hai...v.v. Việc xây dựng cấu trúc h_menu như thế ta có thể hiểu là đây là một danh sách liên kết đơn mà mỗi một node của danh sách này lại là một danh sách liên kết đơn chứa node con là menu_items cấp hai. Như vậy việc xây dựng cấu trúc h_menu như thế này sẽ giúp ta dễ dàng trong việc hiển thị lên màn hình và việc thêm bớt các chức năng trong giao diện người dùng.

```
4 #define h_menu H_menu*
5 struct H_menu
6 {
7     TEXT name_h_menu = "";
8     bool active = false;
9     bool houver = false;
10
11    Uint32 x_location = 0;
12    Uint32 y_location = 0;
13
14    int num_sub_menu = 0;
15    int num_next_menu = 0;
16
17    Uint32 h_menu_width = 100;
18    Uint32 h_menu_height = 40;
19
20    w_tab handle_to_wtab = nullptr;
21    SDL_Color h_menu_color = DEFAULT_h_menu_COLOR;
22
23    h_menu next_h_menu = nullptr;
24    h_menu list_sub_menu = nullptr;
25
26};
```

Figure 7. Cấu trúc h_menu

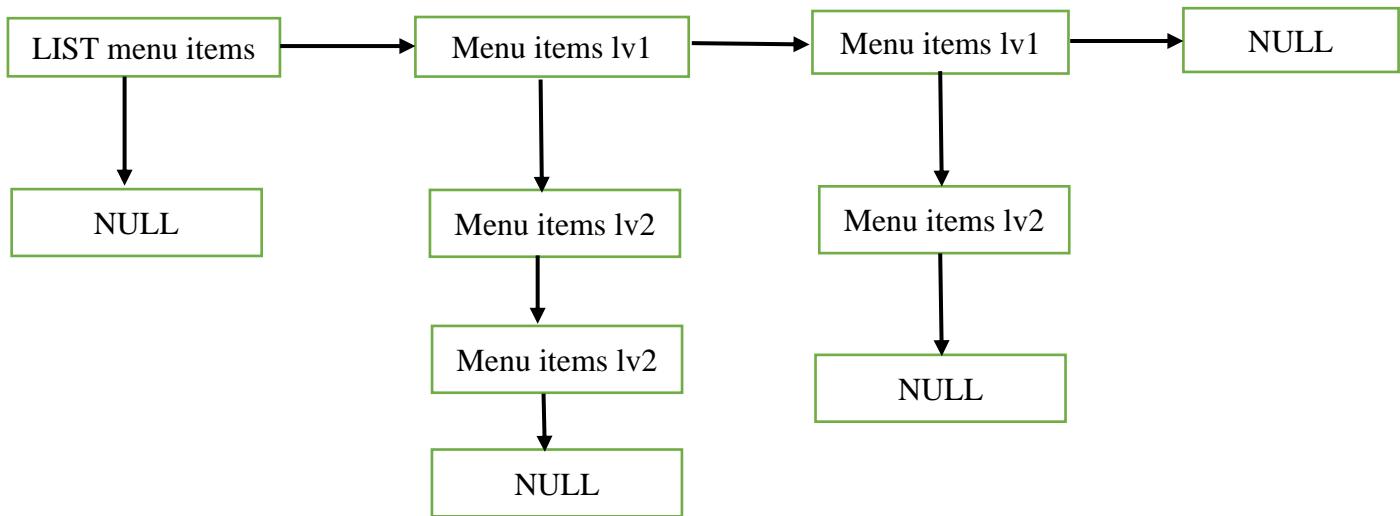
Cấu trúc Menu_bar cấu trúc này thể hiện cho thanh menu bar được hiển thị phía trên màn hình. Các trường dữ liệu của cấu trúc này thể hiện kích thước của thanh menu bar, màu sắc của thanh menu bar, đường dẫn tới list các menu_bar_items ..v.v..

Cấu trúc Window cấu trúc này bao gồm toàn bộ màn hình, Các trường dữ liệu của cấu trúc này thể hiện kích thước màn hình, tên chương trình, tọa độ xuất hiện chương trình, đường dẫn tới thanh menu bar,,v.v.

Cấu trúc w_tab, cấu trúc này sẽ có nhiệm vụ hiển thị thông tin từ các bản ghi lên màn hình. Đây là cấu trúc đặc biệt khi địa chỉ của cấu trúc này được lưu bởi cấu trúc h_menu tuy nhiên nó cũng là phần tử con của cấu trúc Window. Khi một đối tượng menu items được chọn hệ thống sẽ chuyển địa chỉ w_tab Của cấu trúc h_menu sang cho cấu trúc Window và được hiển thị lên màn hình.

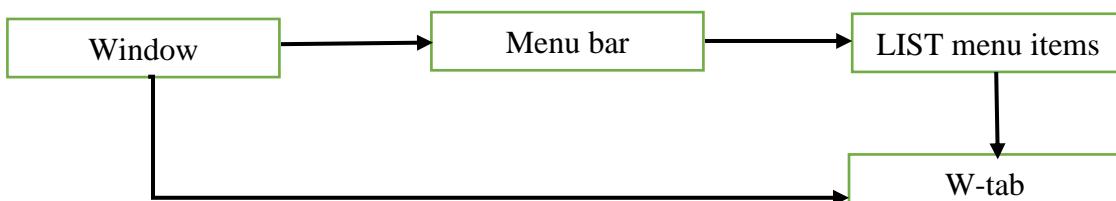
Sơ đồ các đối tượng được thể hiện trong các sơ đồ dưới đây:

Cấu trúc h_menu:



Nhận xét: với việc phân chia cấu trúc menu_items như trên thì rất thuận tiện cho việc vẽ các đối tượng. Đầu tiên ta sẽ tạo hàm vẽ đối tượng là menu items cấp 1 và tạo hàm vẽ đối tượng là menu items cấp 2. Việc vẽ các đối tượng sẽ tiến hành như sau. Thứ nhất chúng ta vẽ đối tượng menu_items gốc sau đó nếu đối tượng đó đang được trỏ bởi chuột thì tiến hành vẽ các menu con cấp 2 bằn hàm vẽ vừa tạo (với điều kiện dừng là con trỏ tới menu con cấp 2 tiếp theo bằng NULL thì dừng). Tiếp tục vẽ các đối tượng menu con cấp 1 cho đến khi gặp con trỏ null thì thôi.

Cấu trúc Window.



Với cấu trúc như thế này cơ bản khi vẽ giao diện người dùng thì ta tiến hành vẽ từ cấu trúc window trước sâu đó tới menu bar rồi tới list menu items. Nếu cấu trúc window có con trỏ tới lớp w_tab khác null thì ta vẽ tiếp cấu trúc w_tab.

Ngoài cấu trúc hiển thị giao diện người dùng như trên ta còn cần các biến toàn cục như: Tọa độ chuột trái, Tọa độ chuột phải, các biến xử lý màu, các biến lưu phòng chữ ..v.v..

2.1.1 Ý tưởng hoàn thành xử lý file.

Trong chương trình này có sử dụng 3 file riêng biệt nên việc xử lý các dữ liệu từ các file này cũng là phức tạp. Vì vậy đối với các file khi được đọc ghi ta cần định nghĩa các cấu trúc lưu trữ riêng để tiện cho công việc đọc ghi file, tránh bị chồng chéo dữ liệu.

Đối với file NV.Bin ta định nghĩa các trường cấu trúc:

<code>"Mã nhân viên"</code><name>"Tên nhân viên"</name><dob>"Ngày sinh"</dob>

<code>: thẻ này định nghĩa trường dữ liệu tiếp theo sẽ là mã nhân viên.

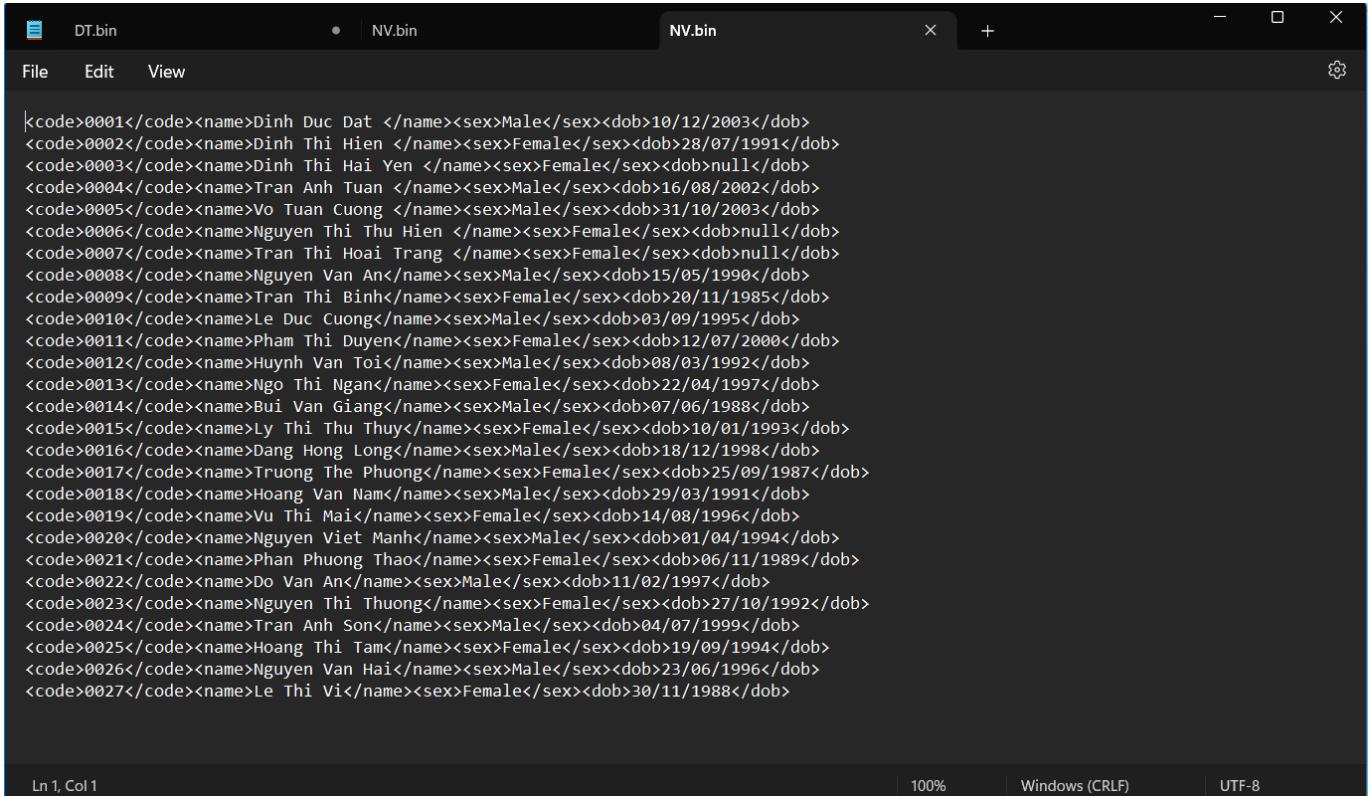
</code>: thẻ này định nghĩa kết thúc trường mã nhân viên.

<name>: thẻ này định nghĩa trường dữ liệu tiếp theo sẽ là tên nhân viên.

</name>: thẻ này định nghĩa kết thúc trường tên nhân viên

<dob>: thẻ này định nghĩa trường dữ liệu tiếp theo sẽ là ngày sinh của nhân viên.

</dob>: thẻ này định nghĩa kết thúc trường tên nhân viên.



```
<code>0001</code><name>Dinh Duc Dat </name><sex>Male</sex><dob>10/12/2003</dob>
<code>0002</code><name>Dinh Thi Hien </name><sex>Female</sex><dob>28/07/1991</dob>
<code>0003</code><name>Dinh Thi Hai Yen </name><sex>Female</sex><dob>null</dob>
<code>0004</code><name>Tran Anh Tuan </name><sex>Male</sex><dob>16/08/2002</dob>
<code>0005</code><name>Vu Tuan Cuong </name><sex>Male</sex><dob>31/10/2003</dob>
<code>0006</code><name>Nguyen Thi Thu Hien </name><sex>Female</sex><dob>null</dob>
<code>0007</code><name>Tran Thi Hoai Trang </name><sex>Female</sex><dob>null</dob>
<code>0008</code><name>Nguyen Van An</name><sex>Male</sex><dob>15/05/1990</dob>
<code>0009</code><name>Tran Thi Binh</name><sex>Female</sex><dob>20/11/1985</dob>
<code>0010</code><name>Le Duc Cuong</name><sex>Male</sex><dob>03/09/1995</dob>
<code>0011</code><name>Pham Thi Duyen</name><sex>Female</sex><dob>12/07/2000</dob>
<code>0012</code><name>Huynh Van Toi</name><sex>Male</sex><dob>08/03/1992</dob>
<code>0013</code><name>Ngo Thi Ngan</name><sex>Female</sex><dob>22/04/1997</dob>
<code>0014</code><name>Bui Van Giang</name><sex>Male</sex><dob>07/06/1988</dob>
<code>0015</code><name>Ly Thi Thu Thuy</name><sex>Female</sex><dob>10/01/1993</dob>
<code>0016</code><name>Dang Hong Long</name><sex>Male</sex><dob>18/12/1998</dob>
<code>0017</code><name>Truong The Phuong</name><sex>Female</sex><dob>25/09/1987</dob>
<code>0018</code><name>Hoang Van Nam</name><sex>Male</sex><dob>29/03/1991</dob>
<code>0019</code><name>Vu Thi Mai</name><sex>Female</sex><dob>14/08/1996</dob>
<code>0020</code><name>Nguyen Viet Manh</name><sex>Male</sex><dob>01/04/1994</dob>
<code>0021</code><name>Phan Phuong Thao</name><sex>Female</sex><dob>06/11/1989</dob>
<code>0022</code><name>Do Van An</name><sex>Male</sex><dob>11/02/1997</dob>
<code>0023</code><name>Nguyen Thi Thuong</name><sex>Female</sex><dob>27/10/1992</dob>
<code>0024</code><name>Tran Anh Son</name><sex>Male</sex><dob>04/07/1999</dob>
<code>0025</code><name>Hoang Thi Tam</name><sex>Female</sex><dob>19/09/1994</dob>
<code>0026</code><name>Nguyen Van Hai</name><sex>Male</sex><dob>23/06/1996</dob>
<code>0027</code><name>Le Thi Vi</name><sex>Female</sex><dob>30/11/1988</dob>
```

Figure 8. Cấu trúc file NV.BIN

Như vậy khi đọc dữ liệu ta chỉ cần lọc dữ liệu nằm giữa hai thẻ <>, </> là ta sẽ có được trường dữ liệu mong muốn. Việc ràng buộc dữ liệu ta sẽ thực hiện khi thêm bản ghi vào hệ thống.

Đối với file DT.BIN ta định nghĩa các trường cấu trúc:

<code>: thẻ này định nghĩa trường dữ liệu tiếp theo sẽ là mã dự án.

</code>: thẻ này định nghĩa kết thúc trường dữ liệu mã dự án.

<name>: thẻ này định nghĩa trường dữ liệu tiếp theo sẽ là tên dự án.

</name>: thẻ này định nghĩa kết thúc trường tên dự án.

<begin>: thẻ này định nghĩa bắt đầu trường dữ liệu ngày bắt đầu.

</begin>: thẻ này định nghĩa kết thúc thẻ <begin>

<end>: thẻ này định nghĩa trường dữ liệu tiếp theo là ngày kết thúc dự án.

</end>: kết thúc thẻ <end>.

<cost>: định nghĩa dữ liệu tiếp theo là chi phí dự án.

</cost>: kết thúc thẻ <cost>.

The screenshot shows a code editor window with two tabs: 'DT.bin' and 'NV.bin'. The 'DT.bin' tab is active, displaying a large XML-like document structure. The content consists of multiple entries, each starting with '<code>' and ending with '</cost>'. Each entry contains a name like 'Quan li tai san', a start date ('begin'), an end date ('end'), and a cost value. The names and dates are repeated across many entries, indicating a large dataset.

```

<code>0001</code><name>Quan li tai san</name><begin>12/01/2002</begin><end>12/12/2010</end><cost>100000000</cost>
<code>0002</code><name>Quan li chuo cung ung</name><begin>12/01/2004</begin><end>12/12/2019</end><cost>500000000</cost>
<code>0003</code><name>Hach toan chi phi</name><begin>31/01/2010</begin><end>12/12/2020</end><cost>958000000</cost>
<code>0004</code><name>Quan li tai san cong</name><begin>12/01/2002</begin><end>12/12/2010</end><cost>198000000</cost>
<code>0005</code><name>Xay dung chuo quan li</name><begin>12/01/2002</begin><end>12/12/2010</end><cost>9200000000</cost>
<code>0006</code><name>Quan li tai san</name><begin>12/01/2002</begin><end>12/12/2010</end><cost>1210000000</cost>
<code>0007</code><name>Quan li tai san tu nhan</name><begin>12/01/2002</begin><end>12/12/2010</end><cost>1010000000</cost>
<code>0008</code><name>Quan li chi phi</name><begin>12/01/2002</begin><end>12/12/2010</end><cost>7010000000</cost>
<code>0009</code><name>Quan li nhan su</name><begin>15/02/2005</begin><end>28/12/2021</end><cost>3500000000</cost>
<code>0010</code><name>Quan li san xuat</name><begin>20/05/2008</begin><end>10/11/2022</end><cost>6000000000</cost>
<code>0011</code><name>Quan li marketing</name><begin>03/09/2012</begin><end>05/06/2023</end><cost>2000000000</cost>
<code>0012</code><name>Quan li tai chinh</name><begin>28/11/2010</begin><end>15/09/2024</end><cost>4500000000</cost>
<code>0013</code><name>Quan li du an</name><begin>10/03/2014</begin><end>25/08/2025</end><cost>8000000000</cost>
<code>0014</code><name>Quan li chat luong</name><begin>18/06/2016</begin><end>01/12/2026</end><cost>3000000000</cost>
<code>0015</code><name>Quan li ho tro khach hang</name><begin>22/09/2018</begin><end>06/03/2027</end><cost>1500000000</cost>
<code>0016</code><name>Quan li mua hang</name><begin>05/12/2020</begin><end>10/07/2028</end><cost>2500000000</cost>
<code>0017</code><name>Quan li kho</name><begin>15/03/2023</begin><end>25/10/2029</end><cost>4000000000</cost>
<code>0018</code><name>Quan li phan phoi</name><begin>28/06/2025</begin><end>05/05/2030</end><cost>7000000000</cost>
```

Figure 9. Cấu trúc file DT.BIN

Đối với file NVDT:

<name>"Tên nhân viên"</name><project>"Tên đê tài"</project><role>"Vai trò"</role>
 <name>: bắt đầu trùng tên nhân viên.
 </name>: kết thúc trùng tên nhân viên.
 <project>: bắt đầu trùng tên dự án.
 </project>: kết thúc trùng tên dự án.
 <role>: bắt đầu trùng vai trò tên nhân viên.
 </role>: kết thúc trùng vai trò nhân viên.

The screenshot shows a code editor window with one tab: 'NVDT.bin'. The content is a simple XML document with four entries. Each entry contains a name like 'Dinh Thi Hien', a project like 'Quan li tai san', and a role like 'Giam doc'. The structure follows the pattern defined in the previous text.

```

<name>Dinh Thi Hien</name><project>Quan li tai san tu nhan</project><role> Giam doc</role>
<name>Dinh Thi Hien</name><project>Quan li tai san cong</project><role> Tu van giam sat</role>
<name>Dinh Duc Dat</name><project>Quan li tai san</project><role> Chu tich</role>
<name>Nguyen Thi Thu Hien</name><project>Quan li tai san</project><role> Pho giam doc</role>
```

Figure 10. Cấu trúc file NVDT.

Với ràng buộc dữ liệu là tên nhân viên và tên dự án phải có trong file NV.BIN và DT.BIN thì ta sẽ kiểm tra trong quá trình ghi dữ liệu vào file.

Với việc tạo cấu trúc file như trên thì ta đã có thể hoàn thiện hành xử lý file tuy nhiên nếu xử lý file trực tiếp như vậy sẽ gặp nhiều vấn đề nghiêm trọng trong quá trình vận hành lâu dài. Ví dụ nếu ta thêm một

bản ghi thì bản ghi mới sẽ chỉ được lưu ở cuối tập tin. Hoặc nếu ta xóa một bản ghi thì vị trí bản ghi cũ sẽ bị trống mà lâu dài sẽ khiến file bị đầy trong khi dữ liệu được lưu chỉ có các hàng trống. Hoặc khi ta muốn hoàn tác các chức năng thì không thể vì ta đã thực hiện trực tiếp trên file. Để khắc phục hạn chế này ta tiến hành tạo một cấu trúc xử lý và lưu trữ file tạm thời. Cấu trúc này có dạng là một danh sách liên kết đơn lưu trữ các bản ghi của đối tượng cần được xử lý. Cụ thể trong chương trình này sẽ có 3 cấu trúc ứng với 3 file NV, DT, NVDT.

Cấu trúc Employee(Ứng với file NV.BIN).

```
5 // Stack class
6 #define employee Employee*
7
8 struct Employee {
9     std::string emp_code = "";
10    std::string emp_name = "";
11    std::string emp_sex = "";
12    std::string emp_dob = "";
13
14    employee next_record = nullptr;
15
16 };
17
```

Figure 11. Cấu trúc Employee

Cấu trúc Employee gồm các trường:

Emp_code: mã nhân viên.

Emp_name: tên nhân viên.

Emp_sex: giới tính nhân viên.

Emp_dob: ngày sinh nhân viên.

Next_record: bản ghi kế tiếp.

Việc lưu dữ liệu vào một cấu trúc dạng danh sách liên kết đơn giúp ta đơn giản hóa quá trình lưu trữ và xử lý dữ liệu so với việc trực tiếp xử lý trên file. Nếu ta thêm bớt các bản ghi thì ta chỉ cần thực hiện theo nguyên tắc thêm bớt trên danh sách liên kết đơn rồi từ đó mới lưu vào các file.

Đối với các file DT.BIN và NVDT.BIN ta cũng xử lý tương tự.

2.2 Sơ đồ khái niệm

Chương trình này được chia thành hai khái niệm chính là xử lý dữ liệu và hiển thị giao diện người dùng. Đối với việc xây dựng hai khái niệm này đã được trình bày ở mục 2.1. Trong mục 2.2 sẽ là sơ đồ khái niệm của chương trình sau khi hoàn thành xây dựng hai khái niệm này.

Trong chương trình này các hàm có chung chức năng sẽ được gộp lại thành các header riêng để tiện cho việc xử lý file sau này.

Các header được tạo:

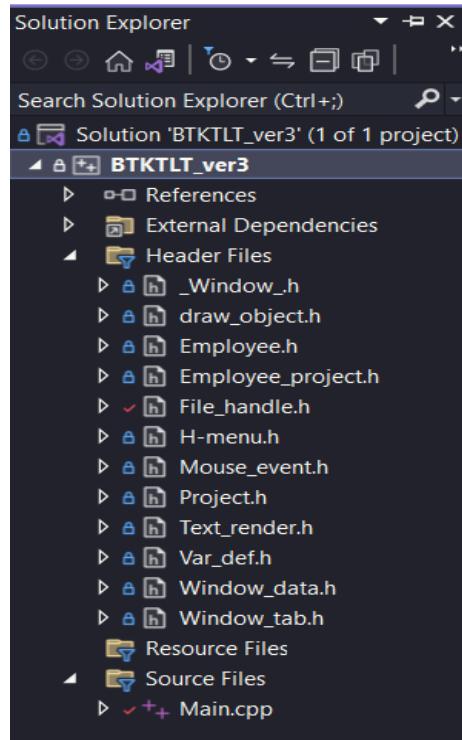


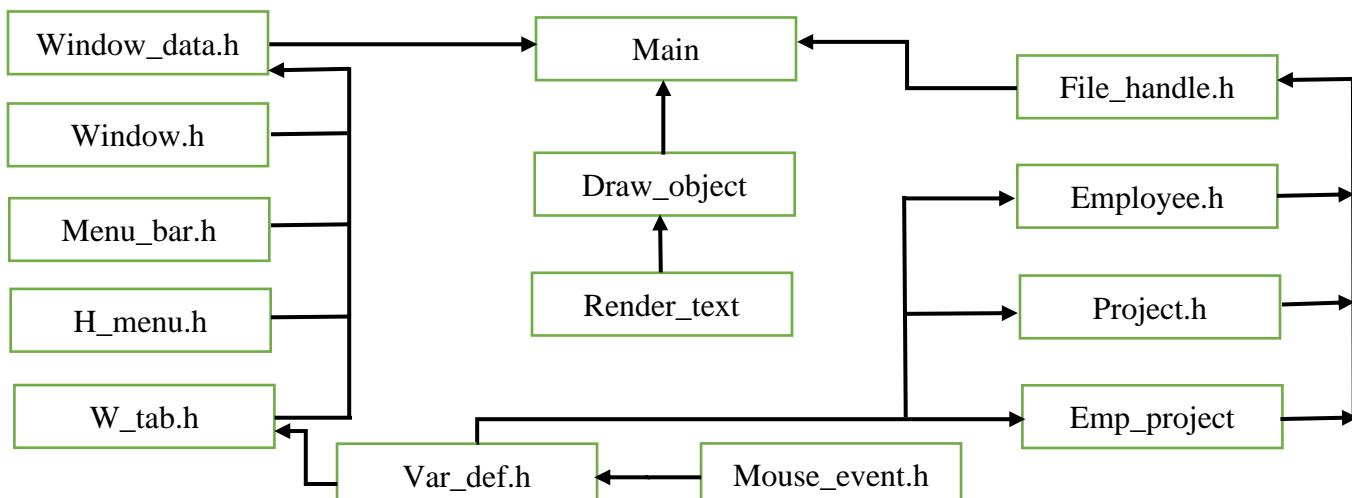
Figure 12. các header được tạo

Các tập tin:

- Var_def.h: header này định nghĩa các biến toàn cục, các biến màu, kích thước cửa sổ mặc định, tọa độ chuột, biến phông chữ, biến lưu thời gian hệ thống, các hàm xử lý thời gian, các hàm xử lý chuỗi.
- draw_object.h: header này định nghĩa các hàm vẽ đối tượng.
- Text_render.h: header này định nghĩa các hàm hiển thị chữ lên màn hình, các hàm hỗ trợ nhập xuất dữ liệu.
- Mouse_event.h: header này định nghĩa các hàm xử lý sự kiện chuột.
- Window_data.h: header này định nghĩa cấu trúc của window trong header này chưa dữ liệu tĩnh của các cấu trúc employee, project, employee_project.
- File_handle.h: header này định nghĩa các hàm xử lý nhập xuất dữ liệu từ các file NV, NVDT, DT và truyền nó vào danh sách employee, employee_project, project.

Các header còn lại là các header định nghĩa các cấu trúc và hàm xử lý đối với các cấu trúc đó như đã trình bày ở mục 2.1

Sơ đồ chương trình.



2.3 Các thuật toán và cấu trúc dữ liệu sử dụng trong chương trình.

2.3.1 Các cấu trúc dữ liệu sử dụng trong chương trình.

Như đã giới thiệu ở các phần trên da phần các cấu trúc dữ liệu sử dụng trong chương trình là dạng danh sách liên kết đơn. Các cấu trúc dữ liệu này được sử dụng để mô hình hóa các thực thể là các bản ghi hoặc các hiển thị giao diện người dùng. Trong phần này ta sẽ phân tích cụ thể hơn các cấu trúc dữ liệu được sử dụng đối với từng thực thể riêng biệt.

1. Cấu trúc window.

```
struct Window
{
    Uint32 x_location = 0;
    Uint32 y_location = 0;

    Uint32 window_width = Default_screen_width;
    Uint32 window_height = Default_screen_height;

    SDL_Color window_color = BLACK;
    menu_bar menu = nullptr;
    w_tab window_tab = nullptr;
};
```

Figure 13. Cấu trúc window

Cấu trúc window được định nghĩa trong header _Window_.h sở dĩ không đặt là Window.h để thuận tiện hơn cho việc lập trình là bởi vì nếu đặt là Window.h sẽ xung đột với header của hệ thống. Nên ta đặt tên header này là _window_.h.

Các biến của cấu trúc này là:

Uint32 x_location : là kiểu số nguyên 32bit không có dấu mang ý nghĩa là hoành độ xuất hiện của chương trình khi chương trình được khởi tạo.

Uint32 y_location: là kiểu số nguyên 32 bit không có dấu mang ý nghĩa là tung độ xuất hiện của chương trình khi chương trình được khởi tạo.

Uint32 window_width: Xác định chiều dài của màn hình khi chương trình được khởi tạo. Mặc định biến này được đặt là Default_screen_width(là biến toàn cục được định nghĩa trong tệp tiêu đề Var_def.h với giá trị là (Uint32)1240).

Uint32 window_height: Xác định chiều rộng của màn hình khi chương trình được khởi tạo. Mặc định biến này được đặt là Default_screen_height(là biến toàn cục được định nghĩa trong tệp tiêu đề Var_def.h với giá trị là (Uint32)1240).

SDL_Color window_color: Màu của màn hình cửa sổ theo hệ RGBA (r,g,b, α) trong đó $0 < r,g,b, \alpha < 256$. Màu mặc định của cửa sổ là màu đen.

Menu_bar menu: con trỏ tới cấu trúc menu_bar được đặt mặc định là nullptr(con trỏ rỗng).

W_tab window_tab con trỏ tới cấu trúc w_tab thể hiện màn hình hiển thị dữ liệu từ file của chương trình, giá trị được đặt mặc định là nullptr.

2. Cấu trúc menu_bar

```
struct Menu_bar
{
    Uint32 x_location = 0;
    Uint32 y_location = 0;
    Uint32 menu_bar_width = default_menu_bar_width;
    Uint32 menu_bar_height = default_menu_bar_height;

    h_menu list_menu_bar_items = nullptr;
};

void add_items_list_to_menu_bar(menu_bar handle_to_menu_bar, h_menu list_menu_bar_items){
    handle_to_menu_bar->list_menu_bar_items = list_menu_bar_items;
}
```

Figure 14. Cấu trúc menu_bar

Cấu trúc menu_bar được định nghĩa chung cùng với cấu trúc window trong tập tiêu đề _Window_.h bởi vì tập các hàm và phương thức của cả hai cấu trúc này khác nhau nên ta gộp cả hai cấu trúc này thành một để tiện theo dõi và xử lý.

Các biến của cấu trúc này là:

Uint32 x_location = 0 : Hoành độ xuất hiện của thanh menu trên màn hình cửa sổ trong suốt chương trình.
Uint32 y_location = 0 : Tung độ xuất hiện của thanh menu trên màn hình cửa sổ trong suốt chương trình.
Uint32 menu_bar_width: chiều dài của thanh menu_bar khi hiện lên màn hình cửa sổ. Giá trị mặc định của nó là default_menu_bar_width = (Uint32)1024 chính là chiều dài của cửa sổ màn hình.
Uint32 menu_bar_height: chiều rộng của thanh menu_bar khi hiện lên màn hình cửa sổ. Giá trị mặc định của nó là default_menu_bar_height = (Uint32)20 chính là chiều dài của cửa sổ màn hình.
H_menu lisst_menu_bar_items: con trỏ trỏ tới danh sách các menu_items được đặt mặc định là con trỏ null_ptr.

2. Cấu trúc h_menu.

```
#define h_menu H_menu*
struct H_menu
{
    TEXT name_h_menu = "";
    bool active = false;
    bool houver = false;

    Uint32 x_location = 0;
    Uint32 y_loaction = 0;

    int num_sub_menu = 0;
    int num_next_menu = 0;

    Uint32 h_menu_width = 100;
    Uint32 h_menu_height = 40;

    w_tab handle_to_wtab = nullptr;
    SDL_Color h_menu_color = DEFAULT_h_menu_COLOR;

    h_menu next_h_menu = nullptr;
    h_menu list_sub_menu = nullptr;
};
```

Figure 15.Cấu trúc h_menu

Cấu trúc h_menu được định nghĩa trong tập tiêu đề h_menu.h. Đây là cấu trúc quan trọng thể hiện các tùy chọn của người dùng trên hệ thống. Cấu trúc này thể hiện các nút nhấn dưới dạng bao gồm một hình chữ nhật và có tệp tùy chọn ở phía trong.

Các biến của cấu trúc này là:

TEXT name_h_menu: là kiểu const_char* thể hiện tên của tùy chọn. Tên này sẽ được hiển thị lên thanh menu. Mặc định tên này là “ ” (rỗng), nên khi ta tiến hành tạo một nút h_menu ta cần khởi tạo luôn tên của nút.

Bool active = false : Biến active có ý nghĩa là nút đã được trỏ bởi con trỏ chuột. Biến này sẽ được trả về giá trị true khi kiểm tra tọa độ nhấn chuột trùng với tọa độ của nút và trả về false nếu tọa độ chuột không trùng với tọa độ của nút sau mỗi lần nhấn chuột. Khi active = true thì chương trình sẽ hiểu là nút đang được chọn và tiến hành vẽ nút này đậm màu hơn các nút khác sau đó vẽ các menu con cấp 2(nếu có) của nút này. Mỗi một thời điểm thì chỉ có một nút được chọn duy nhất.

Bool houver = false : Biến này thể hiện vị trí của chuột có đang nằm trong nút hay không. Nếu có thì vẽ đậm màu hơn.

Int num_sub_menu: biến này thể hiện số lượng các menu con cấp hai của một nút là nút gốc cấp 1. Đối với các nút là nút con cấp 2 thì tham số này luôn bằng 0.

Int num_next_menu: biến này có ý nghĩa đối với nút gốc của danh sách thể hiện số phần tử con là nút con cấp 1 của danh sách.

Nhận xét: ý nghĩa của num_sub_menu và num_next_menu thực tế không phải là chỉ để biết số lượng các nút con của nút gốc mà ý nghĩa thực tế của hai tham số này là để khi ta thêm một nút vào danh sách ta sẽ dựa vào hai tham số này để đánh tọa độ cho các nút con.

Uint32 h_menu_width: chiều dài của nút menu.

Uint32 h_menu_height: chiều rộng của nút menu.

W_tab handle_to_wtab: Đường dẫn tới cấu trúc w_tab khi một nút có giá trị active được chuyển thành true thì đường dẫn này sẽ được chuyển tới cấu trúc window và được hiển thị lên màn hình.

H_menu next_h_menu: Con trỏ tới node con cấp 1 kế tiếp mặc định là nullptr.

H_menu next_sub_menu: Con trỏ tới node con cấp 2 kế tiếp mặc định là nullptr.

3. Cấu trúc w-tab.

```
#define w_tab Window_tab*
#define input_text_box Input_text_box*
struct Window_tab {
    std::vector<std::string> line_display;
    Uint32 line_display_width = default_menu_bar_width;
    Uint32 line_display_height = 30;

    SDL_Color line_color;
};
```

Figure 16. Cấu trúc w_tab

Cấu trúc w_tab định nghĩa trong tập tin W_tab.h. Cấu trúc này thể hiện thông tin từ file dữ liệu lên màn hình chương trình.

Các biến của cấu trúc này là:

Std::vector<std::string> line_display: line display là một vector thể hiện các dòng trên màn hình. Mỗi dòng có độ cao là line_display_height và chiều dài là line_display_width.

4. Cấu trúc Employee.

```
//stack list
#define employee Employee*

struct Employee {
    std::string emp_code = "";
    std::string emp_name = "";
    std::string emp_sex = "";
    std::string emp_dob = "";

    employee next_record = nullptr;
};
```

Figure 17. Cấu trúc Employee.

Cấu trúc Employee lưu trữ dữ liệu vào bộ nhớ tạm. Đây là một cấu trúc danh sách liên kết đơn lưu trữ các bản ghi tạm. Cấu trúc employee sẽ được khởi tạo mỗi khi chương trình được khởi tạo bằng cách đọc từ file và gán nó vào cấu trúc.

Các biến của cấu trúc employee:

Std::string emp_code: mã nhân viên.

Std::string emp_name: tên nhân viên.

Std::string emp_sex: giới tính nhân viên.

Std::string emp_dob: ngày sinh của nhân viên ở đây là kiểu string tuy nhiên trong thực tế thời gian phải được kiểm tra xem có hợp lệ hay không. Cho nên mỗi khi nhập vào trường dữ liệu ngày sinh ta cần phải kiểm tra lại trường dữ liệu này bằng cách chuyển từ std::string sang kiểu int sau đó mới thực hiện

phép toán kiểm tra. Trong trường hợp này ta để emp_dob là kiểu std::string vì để có thể dễ dàng hiển thị lên màn hình chương trình.

Employee next_record: con trỏ tới bản ghi tiếp theo.

Như vậy ta đã xây dựng hoàn tất cấu trúc employee để lưu trữ dữ liệu tạm từ file. Cấu trúc employee có dạng là một danh sách liên kết đơn, ta có thể thực hiện thêm bớt các bản ghi trên danh sách liên kết đơn này bằng các giải thuật tương tự như giải thuật thêm bớt trên danh sách liên kết đơn. Sau khi quá trình hoàn tất ta tiến hành định dạng lại file để xóa hết dữ liệu cũ trong file và tiến hành lưu thông tin từ bộ nhớ tạm vào trong file. Bằng cách này cấu trúc file trong suốt quá trình thực hiện chương trình luôn ổn định và không bị phân mảnh. Tuy nhiên đối với cách làm này đổi lại ta phải trả giá bằng thời gian thực hiện chương trình so với cách thao tác trực tiếp trên file.

5. Cấu trúc Project.

```
#define project Project*
struct Project {
    std::string project_code = "";
    std::string project_name = "";
    std::string year_begin = "";
    std::string year_end = "";
    std::string project_cost = "";

    project next_project_record = nullptr;
};
```

Figure 18. Cấu trúc project.

Cấu trúc project cũng tương tự cấu trúc employee là một cấu trúc dạng danh sách liên kết đơn. Nó có chức năng tương tự như cấu trúc employee trong việc lưu trữ dữ liệu vào bộ nhớ tạm. Việc duyệt bản ghi trên cấu trúc này thực hiện tương tự các giải thuật trên danh sách liên kết đơn.

6. Cấu trúc Emp-prj.

```
#define emp_prj Emp_Prj*
struct Emp_Prj {
    std::string project_name = "";
    std::string employ_name = "";
    std::string role = "";

    emp_prj next_record = nullptr;
};
```

Figure 19. Cấu trúc emp_prj

Cấu trúc này có chức năng tương tự như hai cấu trúc đã trình bày ở trên. Tuy nhiên việc đảm bảo ràng buộc dữ liệu đối với hai cấu trúc phía bên trên sẽ yêu cầu ta phải thực hiện duyệt hai cấu trúc trên trước khi ta thao tác trên cấu trúc emp_prj.

2.3 Giải thuật mà chương trình sử dụng.

Trong chương trình này sử dụng khá nhiều giải thuật dưới đây là một số giải thuật tiêu biểu:

- Giải thuật tìm kiếm trên danh sách liên kết đơn.
- Giải thuật sắp xếp trên danh sách liên kết đơn.
- Giải thuật duyệt danh sách liên kết đơn...v.v

2.4 mã nguồn của chương trình.

Chương trình này được chia thành khá nhiều tập tiêu đề như đã trình bày ở mục 2. Dưới đây là mã nguồn cụ thể và phân tích sâu hơn về các tập tiêu đề của chương trình.

1. Tập tiêu đề Var_def.h

Tập tiêu đề Var_def.h định nghĩa các biến toàn cục, biến điều khiển, các hàm xử lý hệ thống. Đây là tập tiêu đề quan trọng trong thư viện đối. Tập tiêu đề này còn giúp các đối tượng, các hàm trong các tập tiêu đề khác nhau giao tiếp với nhau.

```

10  ifndef TEXT
11  define TEXT const char*
12  endif // !TEXT
13
14
15  static std::string time_system = "";
16  static std::string date_system = "";
17  static std::string log_string_1 = " ";
18  static std::string log_string_2 = "Save file Succeed !! ";
19
20  void get_log_string_1() {
21  |    log_string_1 = "Time: " + time_system + " " + date_system + " : ";
22  }
23  static int x_mouse_location = -1;
24  static int y_mouse_location = -1;
25  static int x_mouse_clickL = -1;
26  static int y_mouse_clickL = -1;
27  static int x_mouse_clickR = -1;
28  static int y_mouse_clickR = -1;
29
30  static TTF_Font* Time_13 = nullptr;
31  static TTF_Font* Time_15 = nullptr;
32  static TTF_Font* Time_17 = nullptr;
33  static TTF_Font* Time_19 = nullptr;
34  static TTF_Font* Time_21 = nullptr;
35  static TTF_Font* Time_22 = nullptr;
36  static TTF_Font* Time_23 = nullptr;
37  static TTF_Font* Time_24 = nullptr;
38
39  static TTF_Font* Consola_7= nullptr;
40  static TTF_Font* Consola_10 = nullptr;
41  static TTF_Font* Consola_13 = nullptr;
42  static TTF_Font* Consola_16 = nullptr;
43  static TTF_Font* Consola_17 = nullptr;

```

Figure 21. Tập tiêu đề Var_def.h pic.1

```

51  static std::string controller = "Null";
52  static std::string controller_s = "Null";
53
54  static std::string input_text = "_";
55  static bool confirm = false;           // dung xong nho tra lai ve false.
56
57  ifndef Default_sceen_width
58  define Default_screen_width (int)1240
59  endif
60  ifndef Default_sceen_height
61  define Default_screen_height (int)720
62  endif
63  define default_menu_bar_width (int)1240
64  define default_menu_bar_height (int)40
65
66  static SDL_Color WHITE = { 255,255,255,255 };
67  static SDL_Color BLACK = { 0,0,0,0 };
68  static SDL_Color BLUE_1 = { 0,255,255,255 };
69  static SDL_Color BLUE_2 = { 153,255,255,255 };
70  static SDL_Color BLUE_3 = { 51,51,85,255 };
71  static SDL_Color BLUE_4 = { 153,153,255,255 };
72  static SDL_Color VIOLET_0 = { 204,153,255,255 };
73  static SDL_Color VIOLET_1 = { 255,153,255,255 };
74  static SDL_Color VIOLET_2 = { 153,0,255,255 };
75  static SDL_Color PINK_0 = { 255,204,109,255 };
76  static SDL_Color PINK_1 = { 255,104,104,255 };
77  static SDL_Color DEFAULT_h_menu_COLOR = { 30,30,30,0 };
78
79  static Uint32 Time_process = 0;
80  static Uint32 Time_start_process = 0;
81  //thoi gian chay he thong.

```

Figure 20. tập tiêu đề Var_def.h pic.2

```

81 //thoi gian chay he thong.
82 char* convertUint32ToChar(Uint32 value)
83 {
84     std::string strValue = std::to_string(value);
85     strValue.append("ms");
86     char* charValue = new char[strValue.length() + 1];
87     strcpy_s(charValue, strValue.length() + 1, strValue.c_str());
88
89     return charValue ;
90 }
91 void get_time_process(Uint32 *time_process) {
92     Uint32 endTime = SDL_GetTicks();
93     *time_process = (endTime - Time_start_process)/10;
94 }
95 std::string get_string_between(std::string start, std::string end, std::string text) {
96     int size_start_str = start.length();
97     int size_end_str = end.length();
98
99     size_t start_a = text.find(start);
100    size_t end_a = text.find(end);
101
102    if (start_a == std::string::npos || end_a == std::string::npos) {
103        return "";
104    }
105    else {
106        start_a += start.length();
107        return text.substr(start_a, end_a - start_a);
108    }
109 }

```

Figure 23. tập tiêu đề Var_def.h pic.3

```

110 void get_time_system() {
111
112     std::time_t currentTime;
113     std::time(&currentTime);
114
115     std::tm localTime;
116     localtime_s(&localTime, &currentTime);
117
118     int date = localTime.tm_mday;
119     int month = localTime.tm_mon;
120     int year = localTime.tm_year;
121     int hour = localTime.tm_hour;
122     int minute = localTime.tm_min;
123     int second = localTime.tm_sec;
124
125     std::string time_sys;
126     time_sys += std::to_string(hour) + ":" + std::to_string(minute) + ":" + std::to_string(second);
127     time_system = time_sys;
128     if (hour <= 11) {
129         time_system += " AM";
130     }
131     else {
132         time_system += " PM";
133     }
134     std::string date_sys = "";
135     date_sys += std::to_string(month + 1) + "/" + std::to_string(date) + "/" + std::to_string(year+1900);
136     date_system = date_sys;
137 }
138

```

Figure 22. tập tiêu đề Var_def.h pic.5

Giải thích:

- Trong tập tiêu đề var_def.h định nghĩa các biến toàn cục, các biến điều khiển chương trình và thực hiện các thao tác liên quan đến các thao tác hệ thống như lấy thời gian thực hệ thống, xử lý, lưu trữ file log.
- Các biến có bắt đầu x_mouse,y_mouse: lưu trữ địa chỉ của con trỏ chuột đối với các thao tác nhấn chuột trái, nhấn chuột phải, di chuyển chuột.
- Các biến SDL_color định nghĩa các màu sắc được hiển thị sử dụng trong cả chương trình.
- Các biến height, width lưu trữ kích thước của các thực thể hiển thị trên màn hình chương trình.
- Các biến year, date, month,... lưu trữ các thông số thời gian hệ thống.

- Các biến TTF_Font* lưu trữ các phông chữ được sử dụng trong chương trình.
- Các biến control: là các biến điều khiển của chương trình. Mặc định khi khởi tạo chương trình các biến này được đặt mặc định là "" tức là không thực hiện bất kỳ thao tác nào.
- Ngoài ra trong tập tiêu đề Vars_def.h còn định nghĩa các hàm lấy thông tin thời gian của hệ thống, định nghĩa các hàm thực hiện chức năng xuất file log của chương trình.

2. Tập tiêu đề text_render.h

```

1 #pragma once
2 #include "Vars_def.h"
3 void init_font_file() {
4     Time_13 = TTF_OpenFont("E:\\times.ttf", 13);
5     Time_15 = TTF_OpenFont("E:\\times.ttf", 15);
6     Time_17 = TTF_OpenFont("E:\\times.ttf", 17);
7     Time_19 = TTF_OpenFont("E:\\times.ttf", 19);
8     Time_21 = TTF_OpenFont("E:\\times.ttf", 21);
9     Time_22 = TTF_OpenFont("E:\\times.ttf", 22);
10    Time_23 = TTF_OpenFont("E:\\times.ttf", 23);
11    Time_24 = TTF_OpenFont("E:\\times.ttf", 24);
12
13
14    Consola_7 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 7);
15    Consola_10 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 10);
16    Consola_13 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 13);
17    Consola_16 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 16);
18    Consola_17 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 17);
19    Consola_18 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 18);
20    Consola_19 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 19);
21    Consola_20 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 20);
22    Consola_21 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 21);
23    Consola_22 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 22);
24    Consola_28 = TTF_OpenFont("E:\\BT_KTLT_III\\BTKTLT_ver3\\consola.ttf", 28);
25 }
26
27 void destroy_font() {
28     TTF_CloseFont(Consola_7);
29     TTF_CloseFont(Consola_10);
30     TTF_CloseFont(Consola_13);
31     TTF_CloseFont(Consola_16);
32     TTF_CloseFont(Consola_17);
33     TTF_CloseFont(Consola_18);
34     TTF_CloseFont(Consola_19);
35     TTF_CloseFont(Consola_20);
36     TTF_CloseFont(Consola_21);
37     TTF_CloseFont(Consola_28);
38     TTF_CloseFont(Time_13);
39
40 }
41 }
```

Figure 25. Tập tiêu đề Text_render.h pic 1

```

42 void drawText(SDL_Renderer* renderer, int x, int y, const char* text, TTF_Font* font)
43 {
44     SDL_Surface* surface;
45     SDL_Texture* texture;
46     SDL_Color textColor = { 255, 255, 255 }; // Đặt màu văn bản là màu xanh (R: 0, G: 0, B: 255)
47
48     surface = TTF_RenderText_Solid(font, text, textColor);
49     texture = SDL_CreateTextureFromSurface(renderer, surface);
50
51     int width = surface->w;
52     int height = surface->h;
53
54     SDL_Rect dstRect = { x, y, width, height };
55     SDL_RenderCopy(renderer, texture, NULL, &dstRect);
56
57     SDL_DestroyTexture(texture);
58     SDL_FreeSurface(surface);
59 }
60
61 void draw_black_Text(SDL_Renderer* renderer, int x, int y, const char* text, TTF_Font* font)
62 {
63     SDL_Surface* surface;
64     SDL_Texture* texture;
65     SDL_Color textColor = { 0, 0, 0 }; // Đặt màu văn bản là màu xanh (R: 0, G: 0, B: 255)
66
67     surface = TTF_RenderText_Solid(font, text, textColor);
68     texture = SDL_CreateTextureFromSurface(renderer, surface);
69
70     int width = surface->w;
71     int height = surface->h;
72
73     SDL_Rect dstRect = { x, y, width, height };
74     SDL_RenderCopy(renderer, texture, NULL, &dstRect);
75
76     SDL_DestroyTexture(texture);
77     SDL_FreeSurface(surface);
78 }
```

Figure 24. Tập tiêu đề Text_render.h pic 2.

- Tập tiêu đề text_render.h định nghĩa các hàm vẽ các đối tượng là chữ lên màn hình chương trình.
- Void init_font_file() hàm này cho phép khởi tạo đường dẫn tới các tập tin đã được định nghĩa trong tập tiêu đề Var_def.h.
- Void destroy_font() hủy các đường dẫn tới các đối tượng font đã được khởi tạo trước đó. Hàm này sẽ được gọi ngay sau khi chương trình hoàn thành để tránh gây rò rỉ bộ nhớ.
- Hàm void draw_text(SDL_renderer,x,y,text,font*) hàm này thực hiện vẽ đối tượng là chữ và hiện thị nó lên màn hình chương trình. Các tham số cho hàm này bao gồm: Thực thể màn hình chương trình, tọa độ x, tọa độ y, chữ cần được hiển thị lên màn hình, đường dẫn tới phông chữ cần được hiển thị lên màn hình chương trình. Mặc định chương trình này sẽ vẽ các đối tượng chữ theo màu trắng.
- Hàm void draw_black_text() các tham số và chức năng tương tự như hàm draw_text tuy nhiên khi chữ được vẽ lên màn hình thì chữ được hiển thị lên màn hình sẽ có màu đen.

3. Tập tiêu đề file_handle.h.

```

7 static employee record_list = new Employee;
8 static project prj_record_list = new Project;
9 static emp_prj e_p_list = new Emp_Prj;
10 bool check_date(std::string data) {
11     return true;
12 }
13 int get_date(std::string data) {
14     int return_value = 0;
15     for (int i = 0; i <= data.size(); i++) {
16     }
17 }
18 return return_value;
19 }

20 void employee_data_solution(std::string data, employee list_record) {
21     std::string code = get_string_between("<code>", "</code>", data);
22     std::string name = get_string_between("<name>", "</name>", data);
23     std::string sex = get_string_between("<sex>", "</sex>", data);
24     std::string dob = get_string_between("<dob>", "</dob>", data);
25     add_record(record_list, create_new_record(code, name, sex, dob));
26 }
27 }

28 void project_data_solution(std::string data, project list_record) {
29     std::string code = get_string_between("<code>", "</code>", data);
30     std::string name = get_string_between("<name>", "</name>", data);
31     std::string begin = get_string_between("<begin>", "</begin>", data);
32     std::string end = get_string_between("<end>", "</end>", data);
33     std::string cost = get_string_between("<cost>", "</cost>", data);
34     add_prj_record(prj_record_list, create_new_prj_record(code, name, begin, end, cost));
35 }
36 }

37 void emp_prj_data_solution(std::string data, emp_prj list_record) {
38     std::string name = get_string_between("<name>", "</name>", data);
39     std::string project_ = get_string_between("<project>", "</project>", data);
40     std::string role = get_string_between("<role>", "</role>", data);
41     add_ep_record(e_p_list, create_new_ep_record(project_, name, role));
42 }

```

Figure 26. Tập tiêu đề file_handle.h.

- Hàm employee_data_solution(string data,employee list_record) hàm này thực hiện thao tác lấy các trường <code>, <name>, <sex>, <dob> từ một đầu vào là string sau đó tạo một bản ghi employee chứa các trường dữ liệu vừa được tạo và thêm nó vào danh sách record chứa các bản ghi nhân viên. Hàm này sẽ được sử dụng để lấy thông tin từ file dưới dạng các chuỗi và chuyển nó thành một danh sách các bản ghi tạm để xử lý.

```

<code>0001</code><name>Dinh Duc Dat </name><sex>Male</sex><dob>10/12/2003</dob>
<code>0002</code><name>Dinh Thi Hien </name><sex>Female</sex><dob>28/07/1991</dob>
<code>0003</code><name>Dinh Thi Hai Yen </name><sex>Female</sex><dob>null</dob>
<code>0004</code><name>Tran Anh Tuan </name><sex>Male</sex><dob>16/08/2002</dob>
<code>0005</code><name>Vo Tuan Cuong </name><sex>Male</sex><dob>31/10/2003</dob>
<code>0006</code><name>Nguyen Thi Thu Hien </name><sex>Female</sex><dob>null</dob>
<code>0007</code><name>Tran Thi Hoai Trang </name><sex>Female</sex><dob>null</dob>
<code>0008</code><name>Nguyen Van An</name><sex>Male</sex><dob>15/05/1990</dob>
<code>0009</code><name>Tran Thi Binh</name><sex>Female</sex><dob>20/11/1985</dob>

```

Figure 27. dữ liệu từ file NV.Bin

- Các hàm project_data_solution(), emp_prj_data_solution() có chức năng tương tự hàm trên.

```

45 void Readfile() {
46
47     std::ifstream inFileNV("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\NV.bin", std::ios::binary);
48     std::ifstream inFileDT("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\DT.bin", std::ios::binary);
49     std::ifstream inFileNDT("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\NVDT.bin", std::ios::binary);
50     if (!inFileNV) {
51         write_log_file(log_string_1 + "Status : Failed opened file NV.bin : Error code: 001FC0\\n");
52         write_log_file(log_string_1 + "Link : 'E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\NV.bin' \\n");
53         return;
54     }
55     else if (!inFileDT) {
56         write_log_file(log_string_1 + "Status : Failed opened file DT.bin : Error code: 001FC0\\n");
57         write_log_file(log_string_1 + "Link : 'E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\DT.bin' \\n");
58         return;
59     }
60     else if (!inFileNDT) {
61         write_log_file(log_string_1 + "Status : Failed opened file NVDT.bin : Error code: 001FC0\\n");
62         write_log_file(log_string_1 + "Link : 'E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\NVDT.bin' \\n");
63         return;
64     }
65     else {
66         write_log_file(log_string_1 + "Status: Open File NV.bin successfully.\\n");
67         write_log_file(log_string_1 + "Status: Open File DT.bin successfully.\\n");
68         write_log_file(log_string_1 + "Status: Open File NVDT.bin successfully.\\n");
69     }
70
71     std::string line;
72     while (std::getline(inFileNV, line)) {
73         employee_data_solution(line, record_list);
74     }
75     while (std::getline(inFileDT, line)) {
76         project_data_solution(line, prj_record_list);
77     }
78     while (std::getline(inFileNDT, line)) {
79         emp_prj_data_solution(line, e_p_list);
80     }
81
82     inFileNV.close();
83     inFileDT.close();

```

Figure 29. tệp tiêu đề file_handle.h pic.2

```

89 void Write_file() {
90     std::ofstream outFileNV("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\test.bin", std::ios::app);
91     std::ofstream outFileDT("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\test.bin", std::ios::app);
92     std::ofstream outFileNDT("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\test.bin", std::ios::app);
93
94     std::string out_line_NV = "";
95     std::string out_line_DT = "";
96     std::string out_line_NDT = "";
97
98     if (outFileNV.is_open()) {
99         employee curr_record = record_list->next_record;
100        while (curr_record->next_record != nullptr) {
101            out_line_NV += <code> + "</code>";
102            out_line_NV += <name> + "</name>";
103            out_line_NV += Search Online + "<sex>";
104            out_line_NV += "<dob>" + curr_record->emp_dob + "</dob>\\n";
105            outFileNV.write(out_line_NV.c_str(), out_line_NV.size());
106
107            curr_record = curr_record->next_record;
108            out_line_NV = "";
109        }
110
111        out_line_NV = "";
112        out_line_NV += "<code>" + curr_record->emp_code + "</code>";
113        out_line_NV += "<name>" + curr_record->emp_name + "</name>";
114        out_line_NV += "<sex>" + curr_record->emp_sex + "</sex>";
115        out_line_NV += "<dob>" + curr_record->emp_dob + "</dob>\\n";
116        outFileNV.write(out_line_NV.c_str(), out_line_NV.size());
117
118        outFileNV.close();
119    }
120    else {
121        std::cout << "Failed to create binary file." << std::endl;
122    }
123
124    if (outFileDT.is_open()) {
125        project curr_record = prj_record_list->next_project_record;
126        while (curr_record->next_project_record != nullptr) {
127            out_line_DT += "<code>" + curr_record->project_code + "</code>";
128        }

```

Figure 28tệp tiêu đề file_handle.h pic.3

```

124
125     if (outFileDT.is_open()) {
126         project curr_record = prj_record_list->next_project_record;
127         while (curr_record->next_project_record != nullptr) {
128             out_line_DT += "<code>" + curr_record->project_code + "</code>";
129             out_line_DT += "<name>" + curr_record->project_name + "</name>";
130             out_line_DT += "<begin>" + curr_record->year_begin + "</begin>";
131             out_line_DT += "<end>" + curr_record->year_end + "</end>";
132             out_line_DT += "<cost>" + curr_record->project_cost + "</cost>\n";
133             outFileDT.write(out_line_DT.c_str(), out_line_DT.size());
134
135             curr_record = curr_record->next_project_record;
136             out_line_DT = "";
137         }
138
139         out_line_DT = "";
140         out_line_DT += "<code>" + curr_record->project_code + "</code>";
141         out_line_DT += "<name>" + curr_record->project_name + "</name>";
142         out_line_DT += "<begin>" + curr_record->year_begin + "</begin>";
143         out_line_DT += "<end>" + curr_record->year_end + "</end>";
144         out_line_DT += "<cost>" + curr_record->project_cost + "</cost>\n";
145         outFileDT.write(out_line_DT.c_str(), out_line_DT.size());
146
147         outFileDT.close();
148     }
149     else {
150         std::cout << "Failed to create binary file." << std::endl;
151     }
152
153
154     if (outFileNVDT.is_open()) {
155         emp_prj curr_record = e_p_list->next_record;
156         while (curr_record->next_record != nullptr) {
157             out_line_NVDT += "<name>" + curr_record->employ_name + "</name>";
158             out_line_NVDT += "<project>" + curr_record->project_name + "</project>";
159             out_line_NVDT += "<role>" + curr_record->role + "</role>\n";
160             outFileNVDT.write(out_line_NVDT.c_str(), out_line_NVDT.size());
161
162             curr_record = curr_record->next_record;
163             out_line_DT = "";
164         }
    }

```

Figure 31. tệp tiêu đề file_handle.h pic.4

```

166     out_line_DT = "";
167     out_line_NVDT += "<name>" + curr_record->employ_name + "</name>";
168     out_line_NVDT += "<project>" + curr_record->project_name + "</project>";
169     out_line_NVDT += "<role>" + curr_record->role + "</role>\n";
170     outFileNVDT.write(out_line_NVDT.c_str(), out_line_NVDT.size());
171
172     outFileDT.close();
173 }
174 else {
175     std::cout << "Failed to create binary file." << std::endl;
176 }
177
178 }

179
180 void Format_file() {
181     std::ofstream outFileNV("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\test.bin", std::ios::binary);
182     std::ofstream outFileDT("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\test.bin", std::ios::binary);
183     std::ofstream outFileNVDT("E:\\BT_KTLT_III\\BTKTLT_ver3\\Data\\test.bin", std::ios::binary);
184
185     outFileNV.close();
186     outFileDT.close();
187     outFileNVDT.close();
188 }

189 void add_employee_record_with_input(std::string input_text, employee record_list) {
190     std::string code = get_string_between("<code>", "</code>", input_text);
191     std::string name = get_string_between("<name>", "</name>", input_text);
192     std::string sex = get_string_between("<sex>", "</sex>", input_text);
193     std::string dob = get_string_between("<dob>", "</dob>", input_text);
194     employee check_record = nullptr;
195     check_record = find_record_by_code(record_list, code);
196
197     if (check_record == nullptr && code != " " && dob != " " && sex != " ") {
198         if (sex == "male" || sex == "female" || sex == "Male" || sex == "Female") {
199             add_record(record_list, create_new_record(code, name, sex, dob));
200             write_log_file(log_string_1 + "Status: Successfully added new employee record : <" + code + ">");
201         }
202     }
203     else {
204         write_log_file(log_string_1 + "Status: Failed added new employee record : <" + code + "> + : Error code: 0001F");
205     }
}

```

Figure 30. tệp tiêu đề file_handle.h pic.5

```

210 }
211 void edit_employee_record(std::string input_text, employee record_list) {
212     std::string code = get_string_between("<code>", "</code>", input_text);
213     std::string name = get_string_between("<name>", "</name>", input_text);
214     std::string sex = get_string_between("<sex>", "</sex>", input_text);
215     std::string dob = get_string_between("<dob>", "</dob>", input_text);
216     employee check_record = nullptr;
217     check_record = find_record_by_code(record_list, code);
218     if (check_record != nullptr && code != " " && dob != " " && sex != " ") {
219         if (sex == "male" || sex == "female" || sex == "Male" || sex == "Female\n") {
220             check_record->emp_name = name;
221             check_record->emp_sex = sex;
222             check_record->emp_dob = dob;
223             write_log_file(log_string_1 + "Status: Successfully edited new employee record : <" + code + ">\n");
224         }
225         else {
226             write_log_file(log_string_1 + "Status: Failed to edit new employee record : <" + code + "> : Error code: 0001F\n");
227         }
228     }
229     else {
230         write_log_file(log_string_1 + "Status: Failed to edit new employee record : <" + code + "> : Error code: 0003F\n");
231     }
232 }
233
234
235 void remove_employee_record(std::string input_text, employee record_list) {
236     std::string code = get_string_between("<code>", "</code>", input_text);
237     std::string name = get_string_between("<name>", "</name>", input_text);
238     std::string sex = get_string_between("<sex>", "</sex>", input_text);
239     std::string dob = get_string_between("<dob>", "</dob>", input_text);
240     employee check_record = nullptr;
241     check_record = find_record_by_code(record_list, code);
242     if (check_record != nullptr) {
243         write_log_file(log_string_1 + "Status: Successfully removed new employee record : <" + code + ">\n");
244         remove_record_bycode(record_list, code);
245     }
246     else {
247         write_log_file(log_string_1 + "Status: Failed removed new employee record : <" + code + ">\n");
248     }

```

Figure 33.tệp tiêu đề file_handle.h pic.6

```

250 void add_new_project(std::string input_text, project prj_list_record) {
251     std::string code = get_string_between("<code>", "</code>", input_text);
252     std::string name = get_string_between("<name>", "</name>", input_text);
253     std::string begin = get_string_between("<begin>", "</begin>", input_text);
254     std::string end = get_string_between("<end>", "</end>", input_text);
255     std::string cost = get_string_between("<cost>", "</cost>", input_text);
256     project check_project = nullptr;
257     check_project = find_prj_record_by_code(prj_list_record, code);
258
259     if (check_project == nullptr && name != " " && begin != " " && end != " " && cost != " ") {
260         add_prj_record(prj_record_list, create_new_prj_record(code, name, begin, end, cost));
261         write_log_file(log_string_1 + "Status: Successfully added new project record : Code: <" + code + ">\n");
262     }
263     else {
264         write_log_file(log_string_1 + "Status: Failed to add new employee record : Code: <" + code + ">\n");
265     }
266
267 }
268
269 void edit_project(std::string input_text, project prj_record_list) {
270     std::string code = get_string_between("<code>", "</code>", input_text);
271     std::string name = get_string_between("<name>", "</name>", input_text);
272     std::string begin = get_string_between("<begin>", "</begin>", input_text);
273     std::string end = get_string_between("<end>", "</end>", input_text);
274     std::string cost = get_string_between("<cost>", "</cost>", input_text);
275     project check_project = nullptr;
276     check_project = find_prj_record_by_code(prj_record_list, code);
277     if (check_project == nullptr && name != " " && begin != " " && end != " " && cost != " ") {
278         check_project->project_name = name;
279         check_project->year_begin = begin;
280         check_project->year_end = end;
281         check_project->project_cost = cost;
282         write_log_file(log_string_1 + "Status: Successfully edited project record : <" + code + ">\n");
283     }
284     else {
285         write_log_file(log_string_1 + "Status: Failed to edit project record : <" + code + ">\n");
286     }
287 }

```

Figure 32.tệp tiêu đề file_handle.h pic.7

```

288 void remove_project(std::string input_text, project prj_record_list) {
289     std::string code = get_string_between("<code>", "</code>", input_text);
290     std::string name = get_string_between("<name>", "</name>", input_text);
291     std::string begin = get_string_between("<begin>", "</begin>", input_text);
292     std::string end = get_string_between("<end>", "</end>", input_text);
293     std::string cost = get_string_between("<cost>", "</cost>", input_text);
294     project check_project = nullptr;
295     check_project = find_prj_record_by_code(prj_record_list, code);
296     if (check_project != nullptr) {
297         write_log_file(log_string_1 + "Status: Successfully removed employee record : Code: <" + code + ">\n");
298         remove_prj_record_bycode(prj_record_list, code);
299     }
300     else {
301         write_log_file(log_string_1 + "Status: Failed to remove project record : Code: <" + code + ">\n");
302     }
303 }
304 void add_ep(std::string input_text, emp_prj ep_lisst) {
305     std::string name = get_string_between("<name>", "</name>", input_text);
306     std::string project_ = get_string_between("<project>", "</project>", input_text);
307     std::string role = get_string_between("<role>", "</role>", input_text);
308     project check_project = nullptr;
309     employee check_employee = nullptr;
310     check_project = find_prj_record_by_name(prj_record_list, project_);
311     check_employee = find_record_by_name(record_list, name);
312     if (check_employee != nullptr && check_project != nullptr && role != " ") {
313         write_log_file(log_string_1 + "Status: Sucessfully added employee project record : Name: <" + name + ">\n");
314         add_ep_record(ep_lisst, create_new_ep_record(project_, name, role));
315     }
316     else {
317         write_log_file(log_string_1 + "Status: Failed to add employee project record : Name: <" + name + ">\n");
318     }
319 }
320 void edit_ep(std::string input_text, emp_prj ep_lisst) {
321     std::string name = get_string_between("<name>", "</name>", input_text);
322     std::string project_ = get_string_between("<project>", "</project>", input_text);
323     std::string role = get_string_between("<role>", "</role>", input_text);
324     project check_project = nullptr;
325     employee check_employee = nullptr;
326     check_project = find_prj_record_by_name(prj_record_list, project_);

```

Figure 35.tệp tiêu đề file_handle.h pic.8

```

329 emp_prj check_ep = find_record_by_project_name(ep_lisst, project_);
330
331 if (check_employee != nullptr && check_project != nullptr && check_ep != nullptr) {
332     check_ep->role = role;
333
334     write_log_file(log_string_1 + "Status: Succesfully edited employee-project record : Name: <" + name + ">\n");
335 }
336 else {
337     write_log_file(log_string_1 + "Status: Failed to edit project record : Name: <" + name + ">\n");
338 }
339 }
340
341 void remove_ep(std::string input_text, emp_prj e_p_list) {
342     std::string name = get_string_between("<name>", "</name>", input_text);
343     std::string project_ = get_string_between("<project>", "</project>", input_text);
344     std::string role = get_string_between("<role>", "</role>", input_text);
345     project check_project = nullptr;
346     employee check_employee = nullptr;
347     check_project = find_prj_record_by_name(prj_record_list, project_);
348     check_employee = find_record_by_name(record_list, name);
349
350     emp_prj check_ep = find_record_by_project_name(ep_lisst, project_);
351
352     if (check_ep != nullptr) {
353         write_log_file(log_string_1 + "Status: Succesfully remove employee-project record : Name: <" + name + ">\n");
354         remove_EPrecord_by_name(ep_lisst, check_ep->project_name);
355     }
356     else {
357         write_log_file(log_string_1 + "Status: Failed to edit employee project record : Name: <" + name + ">\n");
358     }
359 }
360
361 void save() {
362     Format_file();
363     Write_file();
364 }

```

Figure 34.tệp tiêu đề file_handle.h pic.9

Trong tập tiêu đề file_handle.h định nghĩa các hàm thao tác với file dữ liệu, cấu trúc bộ nhớ tạm. Các hàm và các chức năng của các hàm:

- Readfile() hàm này đọc dữ liệu từ 3 file là NV.BIN, DT.BIN và NVDT.bin sau đó thực hiện chuyển đổi và xử lý dữ liệu và chuyển nó thành dữ liệu lưu trữ trong bộ nhớ tạm bằng cách sử dụng các hàm data_solution đã được trình bày ở phần trên.

- Writefile() hàm này thực hiện ghi dữ liệu từ bộ nhớ tạm vào các file.BIN.
- Format_file() hàm này thực hiện xóa dữ liệu trong file cũ để tiến hàn lưu trữ dữ liệu mới vào file.
- Save() hàm này đầu tiên sẽ thực hiện xóa dữ liệu trong file cần ghi sau đó thực hiện ghi dữ liệu từ bộ nhớ tạm vào file cần lưu.
- Ngoài ra trong tập tiêu đề file _handle.h còn định nghĩa các hàm thao tác với cấu trúc dữ liệu là danh sách các bản ghi nhân viên, đê tài, nhân viên – đê tài như là thêm bớt xóa sửa, tìm kiếmv.v.

4. Tập tiêu đề project.h, employee.h employee_project.h

Mã nguồn:

```

1  #pragma once
2  #include <iostream>
3  #include <string>
4
5  //stack list
6  #define employee Employee*
7
8  struct Employee {
9      std::string emp_code = "";
10     std::string emp_name = "";
11     std::string emp_sex = "";
12     std::string emp_dob = "";
13
14     employee next_record = nullptr;
15
16 };
17
18 bool check_empty_list(employee head_record) {
19     if (head_record->next_record == nullptr) {
20         return true;
21     }
22     else {
23         return false;
24     }
25 }
26
27 employee create_new_record(std::string code, std::string name, std::string sex, std::string dob) {
28     employee new_record = new Employee();
29
30     new_record->emp_code = code;
31     new_record->emp_name = name;
32     new_record->emp_sex = sex;
33     new_record->emp_dob = dob;
34
35     new_record->next_record = nullptr;
36
37     return new_record;
38 }
39 // add_record in tail of list_record

```

Hình 36. tập tiêu đề employee.h pic.1

```

41
42     void add_record(employee head_record, employee record) {
43         if (check_empty_list(head_record) == true) {
44             head_record->next_record = record;
45         }
46         else {
47             employee curr_record = head_record;
48             while (curr_record->next_record != nullptr) {
49                 curr_record = curr_record->next_record;
50             }
51             curr_record->next_record = record;
52         }
53     }
54
55     void print_list_record(employee head_record) {
56         if (check_empty_list(head_record) == true) {
57             return;
58         }
59         else {
60             employee curr_record = head_record->next_record;
61             while (curr_record->next_record != nullptr) {
62                 std::cerr << curr_record->emp_name << std::endl;
63                 curr_record = curr_record->next_record;
64             }
65             std::cerr << curr_record->emp_name << std::endl;
66         }
67     }
68 }
69 employee find_record_by_code(employee head_record, std::string code) {
70     if (check_empty_list(head_record) == true) {
71         return nullptr;
72     }
73     else {
74         employee curr_record = head_record->next_record;
75         while (curr_record->next_record != nullptr)
76         {
77             if (curr_record->emp_code == code) {
78                 return curr_record;
79             }
80             curr_record = curr_record->next_record;
81         }
82     }

```

Hình 2. tập tiêu đề employee.h pic.2

```

employee find_record_by_name(employee head_record, std::string name) {
    if (check_empty_list(head_record) == true) {
        return nullptr;
    }
    else {
        employee curr_record = head_record->next_record;
        while (curr_record->next_record != nullptr)
        {
            if (curr_record->emp_name == name) {
                return curr_record;
            }
            curr_record = curr_record->next_record;
        }
        if (curr_record->emp_name == name) {
            return curr_record;
        }
        else {
            return nullptr;
        }
    }
}
void remove_record_bycode(employee head_record, std::string code) {
    if (head_record == nullptr) {
        return;
    }
    else {
        employee curr_record = head_record->next_record;
        employee temp = head_record;
        while (curr_record->next_record != nullptr) {
            if (curr_record->emp_code == code) {
                employee temp_2 = curr_record;
                temp->next_record = curr_record->next_record;
                delete temp_2;
                return;
            }
            curr_record = curr_record->next_record;
            temp = temp->next_record;
        }
    }
}

```

Hình 1 tập tiêu đề employee.h pic.3

```

136     void remove_record_byname(employee head_record, std::string name) {
137         if (head_record == nullptr) {
138             return;
139         }
140         else {
141             employee curr_record = head_record->next_record;
142             employee temp = head_record;
143             while (curr_record->next_record != nullptr) {
144                 if (curr_record->emp_name == name) {
145                     employee temp_2 = curr_record;
146                     temp->next_record = curr_record->next_record;
147                     delete temp_2;
148                     return;
149                 }
150                 curr_record = curr_record->next_record;
151                 temp = temp->next_record;
152             }
153             if (temp->next_record->emp_name == name) {
154                 temp->next_record = nullptr;
155                 delete curr_record;
156             }
157         }
158     }
159 }
160 void set_information(employee record_list, std::string code, std::string new_name, std::string new_sex, std::string new_date_o_birth) {
161     employee curr_record = find_record_by_code(record_list, code);
162     curr_record->emp_name = new_name;
163     curr_record->emp_sex = new_sex;
164     curr_record->emp_dob = new_date_o_birth;
165 }
166 void add_employee_record(employee record_list, std::string data_in) {
167 }
168 }
```

Hình 3 tập tiêu đề employee.h pic.4

- Tập tiêu đề employee.h định nghĩa ra một danh sách liên kết đơn trong đó lưu các bản ghi chứa dữ liệu nhân viên được lấy từ file NV.BIN. Các node trong danh sách liên kết đơn sau khi được khởi tạo và đưa vào danh sách sẽ được thêm xóa, sửa theo yêu cầu của người dùng rồi mới tiến hành lưu vào file lưu trữ.
- Hàm check_emptylist() trả về true nếu danh sách là rỗng, false nếu danh sách không phải là danh sách rỗng.
- Hàm find_record_by_name(string name): nhập vào hàm một chuỗi ký tự và chương trình sẽ trả về các node có trường name là chuỗi được nhập vào.
- Hàm find_record_by_code(string code): Nhập vào một mã số nhân viên và chương trình sẽ trả về các node có mã nhân viên là mã được nhập vào.
- Add_record(*node): thêm một node vào danh sách liên kết đơn.
- Remove_record_by_name(string name): xóa các bản ghi có trường name được nhập vào.
- Remove_record_by_code(string code): xóa các bản ghi có trường code được nhập vào.
- Set_information(): sửa đổi dữ liệu của node.

Đối với hai tập tiêu đề còn lại ta cũng thực hiện các cấu trúc và các chức năng tương tự project.h và emp_prj.h

Mã nguồn:

```

1 #pragma once
2 #include <iostream>
3
4 #define project Project*
5 struct Project {
6     std::string project_code = "";
7     std::string project_name = "";
8     std::string year_begin = "";
9     std::string year_end = "";
10    std::string project_cost = "";
11
12    project next_project_record = nullptr;
13 };
```

Hình 4. định nghĩa cấu trúc project trong tập tiêu đề project.h

```

1 #pragma once
2 #include "Employee.h"
3 #include "Project.h"
4
5 #pragma once
6 #include "Employee.h"
7 #include "Project.h"
8
9
10 #define emp_prj Emp_Proj*
11 struct Emp_Proj {
12     std::string project_name = "";
13     std::string employ_name = "";
14     std::string role = "";
15
16     emp_prj next_record = nullptr;
17 };
18
19 bool check_empty_ep_list(emp_prj head_record) {
20     if (head_record->next_record == nullptr) {
21         return true;
22     }
23     else {
24         return false;
25     }
26 }
27
28 emp_prj create_new_ep_record(std::string prj_name, std::string emp_name, std::string role) {
29     emp_prj new_record = new Emp_Proj;
30
31     new_record->project_name = prj_name;
32     new_record->employ_name = emp_name;
33     new_record->role = role;
34
35     new_record->next_record = nullptr;
36
37     return new_record;
38 }
39 // add_record in tail of list_record

```

Hình 5. định nghĩa cấu trúc emp_prj

Phía trên là toàn bộ các tập tiêu đề của chương trình đã được trình bày và giải thích kỹ. dưới đây sẽ là các tập tiêu đề còn lại của mã nguồn.

1. Tập tiêu đề H-menu.h

```

1 #pragma once
2 #include "Window_tab.h"
3
4 #define h_menu H_menu*
5 struct H_menu {
6     TEXT name_h_menu = "";
7     bool active = false;
8     bool houver = false;
9
10    Uint32 x_location = 0;
11    Uint32 y_loaction = 0;
12
13    int num_sub_menu = 0;
14    int num_next_menu = 0;
15
16    Uint32 h_menu_width = 100;
17    Uint32 h_menu_height = 40;
18
19    w_tab handle_to_wtab = nullptr;
20    SDL_Color h_menu_color = DEFAULT_h_menu_COLOR;
21
22    h_menu next_h_menu = nullptr;
23    h_menu list_sub_menu = nullptr;
24
25 };
26
27 h_menu create_h_menu(TEXT name, w_tab window_tab_link) {
28     h_menu curr_h_menu = new H_menu;
29
30     curr_h_menu->name_h_menu = name;
31     curr_h_menu->handle_to_wtab = window_tab_link;
32     return curr_h_menu;
33 }
34 void set_h_menu(h_menu handle_to_h_menu, TEXT name, w_tab window_tab_link) {
35     handle_to_h_menu->name_h_menu = name;
36     handle_to_h_menu->handle_to_wtab = window_tab_link;
37 }
38 void add_sub_menu(h_menu F_hmenu, h_menu next_sub_menu) {
39
40     h_menu curr_menu = F_hmenu;
41     while (true) {

```

Hình 6. tập tiêu đề h_menu.h

```

34 void set_h_menu(h_menu handle_to_h_menu, TEXT name, w_tab window_tab_link) {
35     handle_to_h_menu->name_h_menu = name;
36     handle_to_h_menu->handle_to_wtab = window_tab_link;
37 }
38 void add_sub_menu(h_menu F_hmenu, h_menu next_sub_menu) {
39
40     h_menu curr_menu = F_hmenu;
41     while (true) {
42         if (curr_menu->list_sub_menu == nullptr) {
43             curr_menu->list_sub_menu = next_sub_menu;
44             break;
45         }
46         curr_menu = curr_menu->list_sub_menu;
47     }
48 }
49 F_hmenu->num_sub_menu = F_hmenu->num_sub_menu + 1;
50
51 next_sub_menu->y_loaction = next_sub_menu->y_loaction + next_sub_menu->h_menu_height * F_hmenu->num_sub_menu;
52 next_sub_menu->x_location = F_hmenu->x_location;
53 }
54 void add_next_h_menu(h_menu handle_to_father_h_menu, h_menu handle_to_next_h_mneu) {
55     h_menu curr_menu = handle_to_father_h_menu;
56     while (true) {
57         if (curr_menu->next_h_menu == nullptr) {
58             curr_menu->next_h_menu = handle_to_next_h_mneu;
59             break;
60         }
61         curr_menu = curr_menu->next_h_menu;
62     }
63 }
64 handle_to_father_h_menu->num_next_menu = handle_to_father_h_menu->num_next_menu + 1;
65 handle_to_next_h_mneu->x_location = handle_to_next_h_mneu->x_location + handle_to_next_h_mneu->h_menu_width * handle_to_father_h_menu->num_next_menu;
66 }
67 void draw_h_menu(h_menu list_items, SDL_Renderer *render) {
68     h_menu curr_menu = list_items;
69     while (true) {
70         if (curr_menu->next_h_menu != nullptr) {
71             if (curr_menu->x_location <= x_mouse_clickL && x_mouse_clickL <= curr_menu->x_location + curr_menu->h_menu_width) {
72                 if (curr_menu->y_loaction <= y_mouse_clickL && y_mouse_clickL <= curr_menu->y_loaction + curr_menu->h_menu_height) {
73                     curr_menu->active = true;
74                     controller = curr_menu->name_h_menu;
75                 }
76             }
77         }
78     }
79 }
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174

```

Hình 7. tập tiêu đề h-menu.h

Tập tiêu đề window_data.h

```

1 #pragma once
2 #include "_Window_.h"
3
4 static window new_window = new Window;
5 static menu_bar new_menu_bar = new Menu_bar;
6
7 static h_menu file = new H_menu;
8 static h_menu file_1 = new H_menu;
9 static h_menu file_2 = new H_menu;
10 static h_menu file_3 = new H_menu;
11 static h_menu file_4 = new H_menu;
12 static h_menu file_5 = new H_menu;
13 static h_menu file_6 = new H_menu;
14
15 static h_menu file_null = new H_menu;
16
17 static h_menu _oposition_1 = new H_menu;
18 static h_menu _oposition_2 = new H_menu;
19 static h_menu _oposition_3 = new H_menu;
20 static h_menu _oposition_4 = new H_menu;
21 static h_menu _oposition_5 = new H_menu;
22
23 static h_menu _oposition_11 = new H_menu;
24 static h_menu _oposition_12 = new H_menu;
25 static h_menu _oposition_13 = new H_menu;
26 static h_menu _oposition_14 = new H_menu;
27 static h_menu _oposition_15 = new H_menu;
28 static h_menu _oposition_16 = new H_menu;
29
30 static h_menu _oposition_21 = new H_menu;
31 static h_menu _oposition_22 = new H_menu;
32 static h_menu _oposition_23 = new H_menu;
33 static h_menu _oposition_24 = new H_menu;
34 static h_menu _oposition_25 = new H_menu;
35 static h_menu _oposition_26 = new H_menu;
36
37 static h_menu _oposition_31 = new H_menu;
38 static h_menu _oposition_32 = new H_menu;
39 static h_menu _oposition_33 = new H_menu;
40 static h_menu _oposition_34 = new H_menu;
41 static h_menu _oposition_35 = new H_menu;

```

Hình 8. tập tiêu đề h-menu.h

```

1 void init_menu() {
2     new_window->window_tab = file_1_tab;
3
4     set_h_menu(file, "File", nullptr);
5     set_h_menu(file_1, "Employee", nullptr);
6     set_h_menu(file_2, "Project", nullptr);
7     set_h_menu(file_3, "Emp Prj", nullptr);
8     set_h_menu(file_4, "Search", nullptr);
9     set_h_menu(file_5, "Help", nullptr);
10    set_h_menu(file_6, "Window", nullptr);
11    set_h_menu(file_null, "null", nullptr);
12
13    set_h_menu(_opsition_1, "Opsition 1", nullptr);
14    set_h_menu(_opsition_2, "Opsition 2", nullptr);
15    set_h_menu(_opsition_3, "Opsition 3", nullptr);
16    set_h_menu(_opsition_4, "Opsition 4", nullptr);
17
18    set_h_menu(_opsition_11, "Add", nullptr);
19    set_h_menu(_opsition_12, "Edit", nullptr);
20    set_h_menu(_opsition_13, "Remove", nullptr);
21    set_h_menu(_opsition_14, "Save", nullptr);
22    set_h_menu(_opsition_15, "Opsition 14", nullptr);
23
24    set_h_menu(_opsition_21, "Add", nullptr);
25    set_h_menu(_opsition_22, "Edit", nullptr);
26    set_h_menu(_opsition_23, "Remove", nullptr);
27    set_h_menu(_opsition_24, "Save", nullptr);
28    set_h_menu(_opsition_25, "Opsition 14", nullptr);
29
30    set_h_menu(_opsition_31, "Add", nullptr);
31    set_h_menu(_opsition_32, "Edit", nullptr);
32    set_h_menu(_opsition_33, "Remove", nullptr);

```

Hình 9. tập tiêu đề window_data.h

Trong tập tiêu đề window_data.h định ta khởi tạo các biến dữ liệu định nghĩa cấu trúc menu tùy chọn và tiến hành thêm vào cách danh sách mặc định. Việc thêm hoặc xóa các chức năng người dùng được tiến hành ở đây một cách dễ dàng và thuận tiện.

2. Tập tiêu đề draw_object.h

```

1 #pragma once
2 #include "Var_def.h"
3
4 void drawSquare(SDL_Renderer* renderer, int x, int y, int size)
5 {
6     SDL_Rect rect = { x, y, size, size };
7     SDL_SetRenderDrawColor(renderer, 255, 0, 0, 255);
8     SDL_RenderFillRect(renderer, &rect);
9 }
10 void drawRectangle(SDL_Renderer* renderer, int x, int y, int width, int height,SDL_Color color)
11 {
12     SDL_Rect rect = { x, y, width, height };
13     SDL_SetRenderDrawColor(renderer, color.r, color.g, color.b, color.a); // Đặt màu vẽ là màu đỏ (R: 255, G: 0, B: 0, Alpha: 255)
14     SDL_RenderFillRect(renderer, &rect);
15 }
16 void drawRoundedSquare(SDL_Renderer* renderer, int x, int y, int size, int cornerRadius, SDL_Color color)
17 {
18
19     SDL_SetRenderDrawColor(renderer, color.r, color.g, color.b, color.a);
20     SDL_Rect outerRect = { x, y, size, size };
21     SDL_RenderDrawRect(renderer, &outerRect);
22
23     int innerSize = size - 2 * cornerRadius;
24     SDL_Rect innerRect = { x + cornerRadius, y + cornerRadius, innerSize, innerSize };
25     SDL_RenderDrawRect(renderer, &innerRect);
26
27     int centerX = x + size / 2;
28     int centerY = y + size / 2;
29
30     SDL_RenderDrawLine(renderer, x + cornerRadius, y, x + size - cornerRadius, y);
31     SDL_RenderDrawLine(renderer, x, y + cornerRadius, x, y + size - cornerRadius);
32     SDL_RenderDrawLine(renderer, x + cornerRadius, y, x, y + cornerRadius);
33
34     SDL_RenderDrawLine(renderer, x + size - cornerRadius, y, x + size, y + cornerRadius);
35     SDL_RenderDrawLine(renderer, x + size, y + cornerRadius, x + size, y + size - cornerRadius);
36     SDL_RenderDrawLine(renderer, x + size - cornerRadius, y + size, x + size, y + size - cornerRadius);
37
38     SDL_RenderDrawLine(renderer, x, y + size - cornerRadius, x + cornerRadius, y + size);
39     SDL_RenderDrawLine(renderer, x, y + size - cornerRadius, x + cornerRadius, y + size - cornerRadius);
40     SDL_RenderDrawLine(renderer, x + cornerRadius, y + size, x, y + size - cornerRadius);

```

Hình 10. tập tiêu đề draw_object.h

Trong tập tiêu đề này định nghĩa các hàm giúp vẽ các đối tượng lên màn hình chương trình người dùng

Toàn bộ mã nguồn được public tại: https://github.com/DinhDat-1012/BTKTLT_ver3.

The screenshot shows a GitHub repository page for 'BTKTLT_ver3'. At the top, there are buttons for 'Pin', 'Unwatch 1', 'Fork 0', and 'Starred 1'. Below the header, it shows 'master' branch, '1 branch', and '0 tags'. There are buttons for 'Go to file', 'Add file', and 'Code'. The main content area shows a single commit by 'DinhDat-1012' titled 'Add project files.' with a timestamp of '3c1594a 3 weeks ago'. The commit message is 'Add project files.' and it has '2 commits'. Below the commit, a list of files is shown:

File	Description	Last Commit
Data	Add project files.	3 weeks ago
.gitattributes	Add .gitattributes and .gitignore.	3 weeks ago
.gitignore	Add .gitattributes and .gitignore.	3 weeks ago
BTKTLT_ver3.sln	Add project files.	3 weeks ago
BTKTLT_ver3.vcxproj	Add project files.	3 weeks ago
BTKTLT_ver3.vcxproj.filters	Add project files.	3 weeks ago
Employee.h	Add project files.	3 weeks ago
Employee_project.h	Add project files.	3 weeks ago
File_handle.h	Add project files.	3 weeks ago
H-menu.h	Add project files.	3 weeks ago
Main.cpp	Add project files.	3 weeks ago
Mouse_event.h	Add project files.	3 weeks ago
Project.h	Add project files.	3 weeks ago
SDL2.dll	Add project files.	3 weeks ago
SDL2_ttf.dll	Add project files.	3 weeks ago

On the right side, there are sections for 'About' (No description, website, or topics provided), 'Activity' (1 star, 1 watching, 0 forks), 'Releases' (No releases published, Create a new release), 'Packages' (No packages published, Publish your first package), 'Languages' (C++ 74.4%, C 25.6%), and 'Suggested Workflows' (Based on your tech stack).

Hình 11. code đã được đẩy lên github

Phân 4. Kiểm thử chương trình.

4.1 Kiểm tra khả năng hoạt động của chương trình.

Code: 0001	Name: Dinh Duc Dat	Sex: Male	Date of Birth: 10/12/2003
Code: 0002	Name: Dinh Thi Hien	Sex: Female	Date of Birth: 28/07/1991
Code: 0003	Name: Dinh Thi Hai Yen	Sex: Female	Date of Birth: null
Code: 0004	Name: Tran Anh Tuan	Sex: Male	Date of Birth: 16/08/2002
Code: 0005	Name: Vo Tuan Cuong	Sex: Male	Date of Birth: 31/10/2003
Code: 0006	Name: Nguyen Thi Thu Hien	Sex: Female	Date of Birth: null
Code: 0007	Name: Tran Thi Hoai Trang	Sex: Female	Date of Birth: null
Code: 0008	Name: Nguyen Van An	Sex: Male	Date of Birth: 15/05/1990
Code: 0009	Name: Tran Thi Binh	Sex: Female	Date of Birth: 20/11/1985
Code: 0010	Name: Le Duc Cuong	Sex: Male	Date of Birth: 03/09/1995
Code: 0011	Name: Pham Thi Duyen	Sex: Female	Date of Birth: 12/07/2000
Code: 0012	Name: Huynh Van Toi	Sex: Male	Date of Birth: 08/03/1992
Code: 0013	Name: Ngo Thi Ngan	Sex: Female	Date of Birth: 22/04/1997
Code: 0014	Name: Bui Van Giang	Sex: Male	Date of Birth: 07/06/1988
Code: 0015	Name: Ly Thi Thu Thuy	Sex: Female	Date of Birth: 10/01/1993
Code: 0016	Name: Dang Hong Long	Sex: Male	Date of Birth: 18/12/1998
Code: 0017	Name: Truong The Phuong	Sex: Female	Date of Birth: 25/09/1987
Code: 0018	Name: Hoang Van Nam	Sex: Male	Date of Birth: 29/03/1991
Code: 0019	Name: Vu Thi Mai	Sex: Female	Date of Birth: 14/08/1996
Code: 0020	Name: Nguyen Viet Manh	Sex: Male	Date of Birth: 01/04/1994
Code: 0021	Name: Phan Phuong Thao	Sex: Female	Date of Birth: 06/11/1989
Code: 0022	Name: Do Van An	Sex: Male	Date of Birth: 11/02/1997
Code: 0023	Name: Nguyen Thi Thuong	Sex: Female	Date of Birth: 27/10/1992
Code: 0024	Name: Tran Anh Son	Sex: Male	Date of Birth: 04/07/1999
Code: 0025	Name: Hoang Thi Tam	Sex: Female	Date of Birth: 19/09/1994
Code: 0026	Name: Nguyen Van Hai	Sex: Male	Date of Birth: 23/06/1996
Code: 0027	Name: Le Thi Vi	Sex: Female	Date of Birth: 30/11/1988

Hình 12. chương trình sau khi khởi tạo

Sau nhiều lần kiểm tra nhận thấy chương trình đã hoạt động ổn định, tin cậy và chính xác. Tuy nhiên còn một số nhược điểm cần khắc phục như là các thao tác người dùng còn tương đối phức tạp, giao diện người dùng còn chưa đẹp mắt. Có thể trong thời gian gần nhất em sẽ tiếp tục thực hiện việc cập nhật và sửa chữa để tài này.

4.2 Kiểm thử các chức năng của hệ thống.

1. Chức năng thêm bản ghi.

Code: 0001	Name: Dinh Duc Dat	Sex: Male	Date of Birth: 10/12/2003
Code: 0002	Name: Dinh Thi Hien	Sex: Female	Date of Birth: 28/07/1991
Code: 0003	Name: Dinh Thi Hai Yen	Sex: Female	Date of Birth: null
Code: 0004	Name: Tran Anh Tuan	Sex: Male	Date of Birth: 16/08/2002
Code: 0005	Name: Vo Tuan Cuong	Sex: Male	Date of Birth: 31/10/2003
Code: 0006	Name: Nguyen Thi Thu Hien	Sex: Female	Date of Birth: null
Code: 0007	Name: Tran Thi Hoai Trang	Sex: Female	Date of Birth: null
Code: 0008	Name: Nguyen Van An	Sex: Male	Date of Birth: 15/05/1990
Code: 0009	Name: Tran Thi Binh	Sex: Female	Date of Birth: 20/11/1985
Code: 0010	Name: Le Duc Cuong	Sex: Male	Date of Birth: 03/09/1995
Code: 0011	Name: Pham Thi Duyen	Sex: Female	Date of Birth: 12/07/2000
Code: 0012	Name: Huynh Van Toi	Sex: Male	Date of Birth: 08/03/1992
Code: 0013	Name: Ngo Thi Ngan	Sex: Female	Date of Birth: 22/04/1997
Code: 0014	Name: Bui Van Giang	Sex: Male	Date of Birth: 07/06/1988
Code: 0015	Name: Ly Thi Thu Thuy	Sex: Female	Date of Birth: 10/01/1993
Code: 0016	Name: Dang Hong Long	Sex: Male	Date of Birth: 18/12/1998
Code: 0017	Name: Truong The Phuong	Sex: Female	Date of Birth: 25/09/1987
Code: 0018	Name: Hoang Van Nam	Sex: Male	Date of Birth: 29/03/1991
Code: 0019	Name: Vu Thi Mai	Sex: Female	Date of Birth: 14/08/1996
Code: 0020	Name: Nguyen Viet Manh	Sex: Male	Date of Birth: 01/04/1994
Code: 0021	Name: Phan Phuong Thao	Sex: Female	Date of Birth: 06/11/1989
Code: 0022	Name: Do Van An	Sex: Male	Date of Birth: 11/02/1997
Code: 0023	Name: Nguyen Thi Thuong	Sex: Female	Date of Birth: 27/10/1992
Code: 0024	Name: Tran Anh Son	Sex: Male	Date of Birth: 04/07/1999
Code: 0025	Name: Hoang Thi Tam	Sex: Female	Date of Birth: 19/09/1994
Code: 0026	Name: Nguyen Van Hai	Sex: Male	Date of Birth: 23/06/1996
Code: 0027	Name: Le Thi Vi	Sex: Female	Date of Birth: 30/11/1988

Hình 13. Thêm một bản ghi vào file Employee.

Sau khi nhấn ok kết quả mà ta thu được:

Quản lý dữ liệu.			
File	Employee	Project	Emp Prj
Search		Help	Window
Code: 0001	Name: Dinh Duc Dat	Sex: Male	Date of Birth: 10/12/2003
Code: 0002	Name: Dinh Thi Hien	Sex: Female	Date of Birth: 28/07/1991
Code: 0003	Name: Dinh Thi Hai Yen	Sex: Female	Date of Birth: null
Code: 0004	Name: Tran Anh Tuan	Sex: Male	Date of Birth: 16/08/2002
Code: 0005	Name: Vo Tuan Cuong	Sex: Male	Date of Birth: 31/10/2003
Code: 0006	Name: Nguyen Thi Thu Hien	Sex: Female	Date of Birth: null
Code: 0007	Name: Tran Thi Hoai Trang	Sex: Female	Date of Birth: null
Code: 0008	Name: Nguyen Van An	Sex: Male	Date of Birth: 15/05/1990
Code: 0009	Name: Tran Thi Binh	Sex: Female	Date of Birth: 20/11/1985
Code: 0010	Name: Le Duc Cuong	Sex: Male	Date of Birth: 03/09/1995
Code: 0011	Name: Pham Thi Duyen	Sex: Female	Date of Birth: 12/07/2000
Code: 0012	Name: Huynh Van Toi	Sex: Male	Date of Birth: 08/03/1992
Code: 0013	Name: Ngo Thi Ngan	Sex: Female	Date of Birth: 22/04/1997
Code: 0014	Name: Bui Van Giang	Sex: Male	Date of Birth: 07/06/1988
Code: 0015	Name: Ly Thi Thu Thuy	Sex: Female	Date of Birth: 10/01/1993
Code: 0016	Name: Dang Hong Long	Sex: Male	Date of Birth: 18/12/1998
Code: 0017	Name: Truong The Phuong	Sex: Female	Date of Birth: 25/09/1987
Code: 0018	Name: Hoang Van Nam	Sex: Male	Date of Birth: 29/03/1991
Code: 0019	Name: Vu Thi Mai	Sex: Female	Date of Birth: 14/08/1996
Code: 0020	Name: Nguyen Viet Manh	Sex: Male	Date of Birth: 01/04/1994
Code: 0021	Name: Phan Phuong Thao	Sex: Female	Date of Birth: 06/11/1989
Code: 0022	Name: Do Van An	Sex: Male	Date of Birth: 11/02/1997
Code: 0023	Name: Nguyen Thi Thuong	Sex: Female	Date of Birth: 27/10/1992
Code: 0024	Name: Tran Anh Son	Sex: Male	Date of Birth: 04/07/1999
Code: 0025	Name: Hoang Thi Tam	Sex: Female	Date of Birth: 19/09/1994
Code: 0026	Name: Nguyen Van Hai	Sex: Male	Date of Birth: 23/06/1996
Code: 0027	Name: Le Thi Vi	Sex: Female	Date of Birth: 30/11/1988
Code: 0028	Name: Dinh Dat	Sex: Male	Date of Birth: 12/01/2003

Hình 14. Hoàn tất thêm một bản ghi vào hệ thống.

2. Xóa một bản ghi khỏi hệ thống.

Quản lý dữ liệu.			
File	Employee	Project	Emp Prj
Search		Help	Window
Code: 0001	Name: Dinh Duc Dat	Sex: Male	Date of Birth: 10/12/2003
Code: 0002	Name: Dinh Thi Hien	Sex: Female	Date of Birth: 28/07/1991
Code: 0003	Name: Dinh Thi Hai Yen	Sex: Female	Date of Birth: null
Code: 0004	Name: Tran Anh Tuan	Sex: Male	Date of Birth: 16/08/2002
Code: 0005	Name: Vo Tuan Cuong	Sex: Male	Date of Birth: 31/10/2003
Code: 0006	Name: Nguyen Thi Thu Hien	Sex: Female	Date of Birth: null
Code: 0007	Name: Tran Thi Hoai Trang	Sex: Female	Date of Birth: null
Code: 0008	Name: Nguyen Van An	Sex: Male	Date of Birth: 15/05/1990
Code: 0009	Name: Tran Thi Binh	Sex: Female	Date of Birth: 20/11/1985
Code: 0010	Name: Le Duc Cuong	Sex: Male	Date of Birth: 03/09/1995
Code: 0011	Name: Pham Thi Duyen	Sex: Female	Date of Birth: 12/07/2000
Code: 0012	Name: Huynh Van Toi	Sex: Male	Date of Birth: 08/03/1992
Code: 0013	Name: Ngo Thi Ngan	Sex: Female	Date of Birth: 22/04/1997
Code: 0014	Name: Bui Van Giang	Sex: Male	Date of Birth: 07/06/1988
Code: 0015	Name: Ly Thi Thu Thuy	Sex: Female	Date of Birth: 10/01/1993
Code: 0016	Name: Dang Hong Long	Sex: Male	Date of Birth: 18/12/1998
Code: 0017	Name: Truong The Phuong	Sex: Female	Date of Birth: 25/09/1987
Code: 0018	Name: Hoang Van Nam	Sex: Male	Date of Birth: 29/03/1991
Code: 0019	Name: Vu Thi Mai	Sex: Female	Date of Birth: 14/08/1996
Code: 0020	Name: Nguyen Viet Manh	Sex: Male	Date of Birth: 01/04/1994
Code: 0021	Name: Phan Phuong Thao	Sex: Female	Date of Birth: 06/11/1989
Code: 0022	Name: Do Van An	Sex: Male	Date of Birth: 11/02/1997
Code: 0023	Name: Nguyen Thi Thuong	Sex: Female	Date of Birth: 27/10/1992
Code: 0024	Name: Tran Anh Son	Sex: Male	Date of Birth: 04/07/1999
Code: 0025	Name: Hoang Thi Tam	Sex: Female	Date of Birth: 19/09/1994
Code: 0026	Name: Nguyen Van Hai	Sex: Male	Date of Birth: 23/06/1996
Code: 0027	Name: Le Thi Vi	Sex: Female	Date of Birth: 30/11/1988
Code: 0028	Name: Dinh Dat	Sex: Male	Date of Birth: 12/01/2003

Hình 15. xóa bản ghi khỏi hệ thống.

Kết quả sau khi thực hiện xóa bản ghi có mã số nhân viên là 0028 là:

	Name:	Sex:	Date of Birth:
Code: 0001	Dinh Duc Dat	Male	10/12/2003
Code: 0002	Dinh Thi Hien	Female	28/07/1991
Code: 0003	Dinh Thi Hai Yen	Female	null
Code: 0004	Tran Anh Tuan	Male	16/08/2002
Code: 0005	Vo Tuan Cuong	Male	31/10/2003
Code: 0006	Nguyen Thi Thu Hien	Female	null
Code: 0007	Tran Thi Hoai Trang	Female	null
Code: 0008	Nguyen Van An	Male	15/05/1990
Code: 0009	Tran Thi Binh	Female	20/11/1985
Code: 0010	Le Duc Cuong	Male	03/09/1995
Code: 0011	Pham Thi Duyen	Female	12/07/2000
Code: 0012	Huynh Van Toi	Male	08/03/1992
Code: 0013	Ngo Thi Ngan	Female	22/04/1997
Code: 0014	Bui Van Giang	Male	07/06/1988
Code: 0015	Ly Thi Thu Thuy	Female	10/01/1993
Code: 0016	Dang Hong Long	Male	18/12/1998
Code: 0017	Tuong The Phuong	Female	25/09/1987
Code: 0018	Hoang Van Nam	Male	29/03/1991
Code: 0019	Vu Thi Mai	Female	14/08/1996
Code: 0020	Nguyen Viet Manh	Male	01/04/1994
Code: 0021	Phan Phuong Thao	Female	06/11/1989
Code: 0022	Do Van An	Male	11/02/1997
Code: 0023	Nguyen Thi Thuong	Female	27/10/1992
Code: 0024	Tran Anh Son	Male	04/07/1999
Code: 0025	Hoang Thi Tam	Female	19/09/1994
Code: 0026	Nguyen Van Hai	Male	23/06/1996
Code: 0027	Le Thi Vi	Female	30/11/1988

Hình 16. hoàn thành xóa bản ghi có mã nhân viên là 0028.

3. Chỉnh sửa bản ghi.

	Name:	Sex:	Date of Birth:
Code: 0001	Dinh Duc Dat	Male	10/12/2003
Code: 0002	Dinh Thi Hien	Female	28/07/1991
Code: 0003	Dinh Thi Hai Yen	Female	null
Code: 0004	Tran Anh Tuan	Male	16/08/2002
Code: 0005	Vo Tuan Cuong	Male	31/10/2003
Code: 0006	Nguyen Thi Thu Hien	Female	null
Code: 0007	Tran Thi Hoai Trang	Female	null
Code: 0008	Nguyen Van An	Male	15/05/1990
Code: 0009	Ngo Thi Binh	Female	20/11/1985
Code: 0010	Le Duc Cuong	Male	03/09/1995
Code: 0011	Pham Thi Duyen	Female	12/07/2000
Code: 0012	Huynh Van Toi	Male	08/03/1992
Code: 0013	Ngo Thi Ngan	Female	22/04/1997
Code: 0014	Bui Van Giang	Male	07/06/1988
Code: 0015	Ly Thi Thu Thuy	Female	10/01/1993
Code: 0016	Dang Hong Long	Male	18/12/1998
Code: 0017	Tuong The Phuong	Female	25/09/1987
Code: 0018	Hoang Van Nam	Male	29/03/1991
Code: 0019	Vu Thi Mai	Female	14/08/1996
Code: 0020	Nguyen Viet Manh	Male	01/04/1994
Code: 0021	Phan Phuong Thao	Female	06/11/1989
Code: 0022	Do Van An	Male	11/02/1997
Code: 0023	Nguyen Thi Thuong	Female	27/10/1992
Code: 0024	Tran Anh Son	Male	04/07/1999
Code: 0025	Hoang Thi Tam	Female	19/09/1994
Code: 0026	Nguyen Van Hai	Male	23/06/1996
Code: 0027	Le Thi Vi	Female	30/11/1988

_<code>0001</code><name>Dinh Van Dat</name><sex>Male</sex><dob>12/12/2000</dob>

ok

5:14:12 AM 8/5/2023

Hình 17. tiến hành chỉnh sửa bản ghi 0001.

Kết quả sau khi tiến hành chỉnh sửa bản ghi 0001.

Quản lý dự án.			
File	Employee	Project	Emp Proj
Search		Help	Window
Code: 0001	Name: Dinh Van Dat	Sex: Male	Date of Birth: 12/12/2000
Code: 0002	Name: Dinh Thi Hien	Sex: Female	Date of Birth: 28/07/1991
Code: 0003	Name: Dinh Thi Hai Yen	Sex: Female	Date of Birth: null
Code: 0004	Name: Tran Anh Tuan	Sex: Male	Date of Birth: 16/08/2002
Code: 0005	Name: Vo Tuan Cuong	Sex: Male	Date of Birth: 31/10/2003
Code: 0006	Name: Nguyen Thi Thu Hien	Sex: Female	Date of Birth: null
Code: 0007	Name: Tran Thi Hoai Trang	Sex: Female	Date of Birth: null
Code: 0008	Name: Nguyen Van An	Sex: Male	Date of Birth: 15/05/1990
Code: 0009	Name: Tran Thi Binh	Sex: Female	Date of Birth: 20/11/1985
Code: 0010	Name: Le Duc Cuong	Sex: Male	Date of Birth: 03/09/1995
Code: 0011	Name: Pham Thi Duyen	Sex: Female	Date of Birth: 12/07/2000
Code: 0012	Name: Huynh Van Toi	Sex: Male	Date of Birth: 08/03/1992
Code: 0013	Name: Ngo Thi Ngan	Sex: Female	Date of Birth: 22/04/1997
Code: 0014	Name: Bui Van Giang	Sex: Male	Date of Birth: 07/06/1988
Code: 0015	Name: Ly Thi Thu Thuy	Sex: Female	Date of Birth: 10/01/1993
Code: 0016	Name: Dang Hong Long	Sex: Male	Date of Birth: 18/12/1998
Code: 0017	Name: Truong The Phuong	Sex: Female	Date of Birth: 25/09/1987
Code: 0018	Name: Hoang Van Nam	Sex: Male	Date of Birth: 29/03/1991
Code: 0019	Name: Vu Thi Mai	Sex: Female	Date of Birth: 14/08/1996
Code: 0020	Name: Nguyen Viet Manh	Sex: Male	Date of Birth: 01/04/1994
Code: 0021	Name: Phan Phuong Thao	Sex: Female	Date of Birth: 06/11/1989
Code: 0022	Name: Do Van An	Sex: Male	Date of Birth: 11/02/1997
Code: 0023	Name: Nguyen Thi Thuong	Sex: Female	Date of Birth: 27/10/1992
Code: 0024	Name: Tran Anh Son	Sex: Male	Date of Birth: 04/07/1999
Code: 0025	Name: Hoang Thi Tam	Sex: Female	Date of Birth: 19/09/1994
Code: 0026	Name: Nguyen Van Hai	Sex: Male	Date of Birth: 23/06/1996
Code: 0027	Name: Le Thi Vi	Sex: Female	Date of Birth: 30/11/1988

Hình 18. sau khi hoàn thành sửa bản ghi.

4.3 Đánh giá mức độ hoàn thiện của chương trình.

Sau khi hoàn thành chương trình nhận thấy chương trình còn nhiều thiếu sót tuy nhiên chương trình đã hoạt động ổn định, và chính xác.