\*. Triển khai logic hệ thống

Day 1: Tìm hiểu yêu cầu về chức năng

* Xem các bản báo cáo của phân tích viên và thiết kế viên về các chức năng cần có của phần mềm.
* Phân tích các chức năng đó và chọn những thuật toán phù hợp để xây dựng.

Day 2: Phân chia công việc

* Lập kế hoạch chi tiết, dự đoán các công việc sẽ có.
* Phân chia nhiệm vụ cho các thành viên trong nhóm.

Day 3: Phân tích thuật toán

* Xác định mục đích của thuật toán: tạo được các chức năng khách hàng yêu cầu.
* Liệt kê các bước cần thực hiện để hoàn thành thuật toán.
* Kiểm tra tính đúng của thuật toán. (kiểm tra biên, kiểm tra độ phức tạp của thuật toán,…)

Day 4: Thiết kế thuật toán

* Chọn phương pháp thiết kế thuật toán.
* Lập kế hoạch chi tiết về cách triển khai các thuật toán.
* Xác định cách viết mã nguồn, ngôn ngữ lập trình, mô hình phát triển phần mềm.

Day 5: Lập bản báo cáo

* Tóm tắt và hoàn thiện những nội dụng, kế hoạch đề ra.
* Tạo bản báo cáo tổng kết các công việc cần phải thực hiện.

\*. Phát triển các thành phần CSDL

- Phân tích yêu cầu

* Từ các bản báo cáo xác định rõ giữa yêu cầu chức năng và phi chức năng.
* Làm rõ hơn về các yêu cầu không rõ ràng, mơ hồ.
* Phân loại dữ liệu
* Xác định dữ liệu cần thu thập, quản lý đáp ứng yêu cầu được của hệ thống.
* Xác định nguồn gốc, chu kỳ cập nhật hệ thống và cách triển khai xử lý dữ liệu.
* Dựa vào các tính chất, mối quan hệ giữa các dữ liệu để phân loại dữ liệu.

. Phân loại dữ liệu theo tính chất và đặc điểm (dữ liệu số: số nguyên, số thập phân, …, văn bản, hình ảnh,…)

. Phân loại dữ liệu theo mối quan hệ(1-n, n-n,…)

* Xác định các thực thể và mối quan hệ giữa chúng
* Sử dụng biểu đồ thực thể - quan hệ (ERD) để mô hình hóa cấu trúc dữ liệu.
* Mô tả thuộc tính của mỗi thực thể, xác định kiểu dữ liệu.
* Chuẩn hóa dữ liệu: phân chia dữ liệu thành các bảng con để tránh sự lặp lại.
* Thiết kế bảng: tạo bảng mối quan hệ, xác định thuộc tính và khóa chính cho từng bảng.
* Xác định cách hệ thống xử lý số liệu(tự động hóa): các quy trình nhập, sửa, xóa, truy xuất từ các bảng.
* Thiết lặp quyền bảo mật cho các đối tượng người dùng khác nhau.
* Kiểm tra đánh giá đảm bảo tính chính xác và hiệu suất của CSDL.
* Lập bản báo cáo.

\*. Lập trình hệ thống

* Phân tích yêu cầu:
* Từ các bản báo cáo xác định lại yêu cầu của phần mềm
* Xây dựng 1 hệ thống cơ bản, khái quát.
* Chọn ngôn ngữ, công cụ lập trình và thư viện.
* Phân chia ứng dụng thành các module, thành phần độc lập để có thể phát triển độc lập với nhau.
* Xác định các biện pháp bảo mật (xác định người dung, kiểm soát truy cập, mã hóa,…)
* Lập trình phần mềm
* Lập trình phần mềm theo các quy tắc trong quá trình viết mã.
* Đặt tên biến, chú thích mã nguồn đây đủ để dễ dàng kiểm tra sửa chữa.
* Kiểm thử tích hợp đảm bảo các module và thành phần hoạt động đúng khi sử dụng phần mềm.

\*. Đánh giá và tối ưu hóa mã nguồn

* Đánh giá mã nguồn
* Kiểm tra tính đúng đắn
* Kiểm tra mã nguồn có gặp phải các lỗi về logic, lỗi biên dịch, lỗi thời gian chạy hay không
* Xác định mã nguồn có hoạt động đúng theo các yêu cầu đề ra.
* Kiểm tra hiệu suất
* Đo thời gian thực thi của mã nguồn để xác định có chạy đúng lịch trình hay không
* Tìm kiếm vùng mã nguồn hoạt động chậm, tốn tài nguyên.
* Kiểm tra bảo mật
* Kiểm tra mã nguồn có lỗ hổng bảo mật hay không.
* Kiểm tra lại phương pháp bảo mật đang sử dụng.
* Kiểm tra chuẩn hóa mã
* Đánh giá mã nguồn có dễ đọc, dễ hiểu, dễ bảo trì, có chú thích và tài liệu hướng dẫn không.
* Đánh giá mã nguồn có tuân thủ các quy tắc đặt tên, hàm, lớp và có tuân theo chuẩn của ngôn ngữ lập trình.
* Kiểm tra tích hợp và kiểm thử
* Kiểm tra sự tích hợp của mã nguồn với các thành phần khác.
* Thực hiện kiểm thử từng phần riêng lẻ và kiểm thử tích hợp.

-- Tối ưu hóa mã nguồn:

- Đơn giản hóa các biểu thức toán học giảm độ phức tạp tăng hiệu suất

* Sử dụng các phếp cộng trừ để tăng tốc độ xử lý của CPU
* Tối ưu hóa việc sử dụng biến tạm
* Tránh sử dụng biến tạm trong các biểu thức tính toán phức tạp để giảm tốc độ tính toán.
* Tách các biểu thức phức tạp thành các biểu thức nhỏ hơn để tránh việc cấp phát vùng nhớ và gán kết quả cuối cùng.
* Sử dụng cấu trúc dữ liệu hiệu quả:
* Chọn cấu trúc dữ liệu phù hợp (ví dụ: danh sách liên kết, cây, bảng băm) để tối ưu việc truy xuất và lưu trữ dữ liệu.
* Giảm thiểu số lần truy vấn CSDL:
* Sử dụng bộ nhớ đệm để lưu trữ dữ liệu đã truy vấn trước đó và tránh truy vấn lại.
* Tối ưu hóa vòng lặp và xử lý chuỗi:
* Tránh sử dụng vòng lặp không cần thiết.
* Sử dụng các hàm thư viện hiệu quả cho xử lý chuỗi.
* Tối ưu hóa mã nguồn
* Loại bỏ các mã nguồn lặp lại.
* Sử dụng các thư viện có sẵn thay vì viết mã.