BÁO CÁO KẾT QUẢ THỬ THÁCH

1. **Nhắc lại để bài**:

Yêu cầu xây dựng một hệ thống quản lý địa chỉ IP với các chức năng CRUD (**C**reate, **R**ead, **U**pdate,**D**elete) cơ bản. Hệ thống sẽ lưu trữ thông tin về các địa chỉ IP, tránh xung đột IP trong quá trình cấp phát cho các máy tính.

**- Mục tiêu:** Làm quen với Golang, xây dựng được REST API cơ bản, làm việc với CSDL MySQL.

**- Yêu cầu API Server cho phép:**

Mức Basic:

+ Thêm địa chỉ IP vào hệ thống.

+ Lấy danh sách tất cả các địa chỉ IP.

+ Xóa một địa chỉ IP khỏi hệ thống.

+ Validate IP (V4/V6)

Mức Advanced:

+ Cấp phát tự động một IP khi yêu cầu

+ Tối ưu tìm kiếm IP

1. **Giới thiệu ý tưởng thiết kế:**

Để thiết kế hệ thống quản lý IP với chức năng CRUD cơ bản như yêu cầu em sẽ phân chia code ra thành 4 thành phần cơ bản gồm:

+ Config: chứa các file khởi chạy cấu hình cho database (MySQL)

+ Handlers: nơi chứa các hàm xử lý logic phục vụ cho những yêu cầu về các chức năng CRUD

+ Models: package chứa file khai báo cấu trúc (struct) của dữ liệu ở đây là địa chỉ IP

+ Utils: lưu các hàm hỗ trợ để xác thực các địa chỉ IP

Cấu trúc dự án:

+ Các tệp chức năng.

* config/db.go: thiết lập kết nối database (MySQL) và các thông số cấu hình.
* models/ip.go: định nghĩa bảng và model trong database.
* handlers/ip\_handler.go: gồm các hàm logic xử lý API (CRUD, validate, cấp phát IP, tối ưu hóa tìm kiếm).
* utils/validate.go: chứa hàm tiện ích như kiểm tra định dạng IP (IPv4/IPv6).

+ Các tệp quan trọng.

* .env: có biến môi trường như thông tin database (DB\_HOST, DB\_USER, DB\_PASS).
* main.go: File chính khởi chạy ứng dụng Golang.
* init.sql: Chứa các câu lệnh SQL để khởi tạo database và các bảng.

+ Các tệp liên quan đến Docker

* Dockerfile: build image Docker cho ứng dụng Golang.
* docker-compose.yml: quản lý, khởi động container MySQL và ứng dụng Golang.
* ip\_management.tar: Docker image đã được export để có thể chạy trên máy khác.

+Tài liệu hướng dẫn

* README.md: Hướng dẫn cách cài đặt và sử dụng dự án.



1. **Trình tự hoạt động của dự án:**

Chạy ứng dụng bằng Docker hoặc trực tiếp trên máy tính:

+ Khi chạy MySQL container, script init.sql được thực thi và tạo bảng ips để lưu trữ cấu trúc dữ liệu địa chỉ IP và khởi tạo các index tối ưu phục vụ truy vấn.

+ Sau khi database được khởi tạo, nó được kết nối thông qua file config/db.go và file biến môi trường .env.

+ Sau khi thiết lập kết nối thành công, API server sẽ chạy thông qua file main.go (file chính) cùng framework gin-gonic/gin, nó chạy trên port 8080, kết nối với database và có các route API phục vụ các thao tác logic.

+ Các request API điều hướng đến file handlers/ip\_handler, file chứa các hàm xử lý logic của từng API.

+ Để phục vụ cùng việc chạy các request API còn có các file phụ như models/ip.go định nghĩa model IP được ánh xạ với bảng ips của MySQL, file utils/validate.go chứa các hàm xác thực hợp lệ của IP hỗ trợ cho handlers/ip\_handler.

+ Dockerfile phục vụ việc đóng gói ứng dụng Golang và Docker image và chạy ở các máy khác. Docker-compose.yml phục vụ chạy toàn bộ hệ thống với MySQL và API, kết nối database với hệ thống qua biến môi tường .env.

Lưu đồ thuật toán:

A diagram of a diagram

AI-generated content may be incorrect.

1. **Giải thích code:**

* Em có phân tích chi tiết trong từng file code.
* Các yêu cầu được đáp ứng trong code:

Mức Basic:

+ Thêm địa chỉ IP vào hệ thống: Hàm AddIP trong file handlers/ip\_handler

+ Lấy danh sách tất cả các địa chỉ IP: Hàm GetAllIPs trong file handlers/ip\_handler

+ Xóa một địa chỉ IP khỏi hệ thống: Hàm DeleteIP trong file handlers/ip\_handler

+ Validate IP (V4/V6): Hàm ValidateIP trong file handlers/ip\_handler và package utils

Mức Advanced:

+ Cấp phát tự động một IP khi yêu cầu: Ham AllocateIP trong file handlers/ip\_handler

+ Tối ưu tìm kiếm IP: 2 câu lệnh tạo chỉ mục ở phần cuối file init.sql được áp dụng khi khởi chạy docker. Phục vụ cho việc truy vấn để xử lý dịch vụ ở các hàm logic trong hanlders/ip\_handler

Các nguồn tham khảo:

* Ý tưởng và code triển khai được lấy từ chatbot Grok và ChatGPT
* Học Syntax Golang: [Go Tutorial](https://www.w3schools.com/go/index.php)
* Kiến thức cơ bản về RESTful API: [RESTful API: Kiến thức thiết yếu cho Backend Developers (phần 1)](https://viblo.asia/p/restful-api-kien-thuc-thiet-yeu-cho-backend-developers-phan-1-MG24Bdd5Lz3)

[Tìm hiểu về RESTful](https://viblo.asia/p/tim-hieu-ve-restful-bWrZnLvQ5xw)