

**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO**

**DỰ ÁN HỆ ĐIỀU HÀNH VÀ MẠNG MÁY TÍNH**

**ĐỀ TÀI:**

*Hệ điều hành:* **Tìm hiểu kĩ thuật lập trình Hook. Xây dựng chương trình hỗ trợ tra từ điển tiếng Anh.**

*Mạng máy tính:* Xây dựng chương trình Client tương tác với Web API

**GIẢNG VIÊN HƯỚNG DẪN**

**PSG.TS Nguyễn Tấn Khôi**

**SINH VIÊN THỰC HIỆN**

**Họ tên Lớp sinh hoạt**

**1. Lê Đức Huy - 102200020 20T1**

**2. Huỳnh Bá Thuận - 102200034 20T1**

*Đà Nẵng, 12/2019*

**MỤC LỤC**

[PHẦN 1: LẬP TRÌNH MẠNG 8](#_Toc27852359)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 9](#_Toc27852361)

[1.1. Bộ giao thức TCP/IP 9](#_Toc27852362)

[1.1.1. Khái niệm 9](#_Toc27852363)

[1.1.2. Các tầng 9](#_Toc27852364)

[1.1.3. Phương thức hoạt động 10](#_Toc27852365)

[1.1.4. Socket và cổng 11](#_Toc27852366)

[1.2. Giao thức HTTP 12](#_Toc27852367)

[1.2.1. Khái niệm 11](#_Toc27852363)

[1.2.2. Mô hình 12](#_Toc27852364)

[1.2.3. Định dạng gói tin 13](#_Toc27852365)

[1.2.4. Phương thức 14](#_Toc27852366)

[1.3. Application Programing Interface 12](#_Toc27852367)

[1.3.1. Khái niệm 15](#_Toc27852368)

[1.3.2. Phương thức hoạt động 15](#_Toc27852369)

[1.3.3. Kiến trúc REST 15](#_Toc27852369)

[1.3.4.JSON 17](#_Toc27852370)

[1.4. Thuật toán chương trình 17](#_Toc27852371)

[1.5. Kết chương 19](#_Toc27852372)

[CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 20](#_Toc27852373)

[2.1. Các chức năng 20](#_Toc27852374)

[2.2. Thiết kế chương trình 20](#_Toc27852374)

[2.3. Sơ đồ luồng dữ liệu 20](#_Toc27852375)

[2.4. Kết chương 20](#_Toc27852378)

[CHƯƠNG 3: TRIỂN KHAI 22](#_Toc27852379)

[3.1. Môi trường cài đặt 22](#_Toc27852380)

[3.1.1. Ngôn ngữ 22](#_Toc27852381)

[3.1.2. Môi trường 22](#_Toc27852382)

[3.1.3. IDE 22](#_Toc27852383)

[3.1.4. Thư viện hỗ trợ 22](#_Toc27852384)

[3.2. Kết quả thực thi chương trình 22](#_Toc27852385)

[3.3. Đánh giá 24](#_Toc27852386)

[3.3.1. Ưu điểm 24](#_Toc27852387)

[3.3.2. Nhược điểm 24](#_Toc27852388)

[3.3.3. Giải pháp đề xuất 24](#_Toc27852389)

[3.4. Kết chương 25](#_Toc27852390)

[PHẦN 2: HỆ ĐIỀU HÀNH 26](#_Toc27852391)

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 27](#_Toc27852394)

[1.1. Cơ sở lý thuyết 27](#_Toc27852395)

[1.1.1. Khái niệm về Hook 27](#_Toc27852396)

[1.1.2. Cơ chế 27](#_Toc27852397)

[1.1.3. Các loại Hook 27](#_Toc27852396)

[1.1.4. Khả năng 27](#_Toc27852397)

[1.1.5. Ứng dụng 27](#_Toc27852396)

[1.1.6. Phạm vi 30](#_Toc27852397)

[1.1.7. Các hàm Win32 cần thiết 30](#_Toc27852396)

[1.1.8. Cài đặt 31](#_Toc27852397)

[1.1.9. Giải phóng 31](#_Toc27852396)

[1.2. Thuật toán chương trình 33](#_Toc27852398)

[1.3. Kết chương 35](#_Toc27852399)

[CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 36](#_Toc27852400)

[2.1. Các chức năng 36](#_Toc27852401)

[2.2. Sơ đồ luồng dữ liệu 36](#_Toc27852402)

[2.3. Kết chương 36](#_Toc27852405)

[CHƯƠNG 3: KẾT QUẢ THỰC THI 37](#_Toc27852407)

[3.1. Môi trường thực thi 37](#_Toc27852408)

[3.1.1. Ngôn ngữ 37](#_Toc27852409)

[3.1.2. Nền tảng 37](#_Toc27852410)

[3.1.3. IDE 37](#_Toc27852411)

[3.2. Kết quả thực nghiệm 38](#_Toc27852412)

[3.3. Đánh giá 40](#_Toc27852421)

[3.3.1. Ưu điểm 40](#_Toc27852422)

[3.3.2. Nhược điểm 40](#_Toc27852423)

[3.4. Hướng phát triển 40](#_Toc27852424)

[3.5. Kết chương 41](#_Toc27852425)

[1. KẾT QUẢ ĐẠT ĐƯỢC 42](#_Toc27852426)

[2. HƯỚNG PHÁT TRIỂN 42](#_Toc27852427)

**DANH SÁCH HÌNH ẢNH**

*Hình 1: Sơ đồ bộ giao thức TCP/IP*

*Hình 2: Phương thức hoạt động của bộ giao thức TCP/IP*

*Hình 3: Mô hình các tầng mạng TCP/IP*

*Hình 4: Mô hình hoạt động của giao thức HTTP*

*Hình 5: Mô hình hoạt động Client-Server sử dụng API*

*Hình 6: Thuật toán làm việc của client*

*Hình 7: Cách thức hoạt động của chương trình*

*Hình 8: Sơ đồ luồng dữ liệu ứng dụng đồng bộ ảnh*

*Hình 9: Xem toàn bộ ảnh*

*Hình 10. Xem chi tiết*

*Hình 11. Chức năng đồng bộ ảnh*

*Hình 12. Chức năng xóa ảnh*

*Hình 13. Giao diện Command Prompt của server*

*Hình 14. Cơ chế của Hook*

*Hình 15. Sơ đồ thuật toán ứng dụng dịch nghĩa*

*Hình 16. Sơ đồ luồng dữ liệu ứng dụng dịch nghĩa*

*Hình 17. Giao diện chính của chương trình*

*Hình 18. Soạn thảo và dịch văn bản*

*Hình 19. Gợi ý từ tiếng anh có nghĩa*

*Hình 20. Dịch toàn window*

**DANH SÁCH TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| **Từ viết tắt** | **Diễn giải** |
| IP | Internet Protocol |
| API | Application Programming Interface |
| URL | Uniform Resource Locator |
| JSON | Javascript Object Notation |
| HTTP | Hypertext Transfer Protocol |

MỞ ĐẦU

**1.**  **Tổng quan về đề tài**

Đồ án các môn học chính của Khoa Công nghệ thông tin nói chung và môn học Cơ sở Ngành Mạng nói riêng nhằm tạo điều kiện cơ bản giúp cho sinh viên:

* *Đi sâu và nắm vững một cách có hệ thống kiến thức đã thu nhận được trong quá trình học lý thuyết, làm bài tập và thực hành.*
* *Từng bước làm quen với các công tác khoa học có định hướng của giáo viên hướng dẫn và hình thành hành vi nghiên cứu độc lập có sự trợ giúp của tài liệu tham khảo.*
* *Gắn quá trình học lý thuyết với công tác nghiên cứu thực tế.*
* *Trình bày rõ ràng và khoa học một vấn đề thuộc lĩnh vực nghiên cứu của mì*nh.

Chính vì lẽ đó mà người làm đồ án môn học phải thực hiện, hoàn thành với kết quả chấp nhận được một khối lượng công việc thuộc lĩnh vực chuyên sâu. Kết quả công việc phản ánh công sức, tài năng, trí tuệ của người làm đồ án và phải được trình bày bằng văn bản trong đồ án môn học theo những chuẩn mực và yêu cầu của giáo viên hướng dẫn. Thời gian cần thiết cho mỗi đồ án là khác nhau, nhưng thời hạn phải hoàn thành là yêu cầu quan trọng đặc biệt. Đồ án môn học phản ánh công sức nghiên cứu nên cần phải được trình bày trước bộ môn và được đánh giá bằng điểm số.

Trong khuôn khổ đồ án, dưới sự hướng dẫn của thầy Nguyễn Tấn Khôi, giảng viên khoa Công nghệ thông tin, Trường Đại học Bách khoa – Đại học Đà Nẵng, em đã nghiên cứu, tìm hiểu và thực hiện hai đề tài “Tìm hiểu kỹ thuật lập trình Hook, xây dựng ứng dụng hỗ trợ tra cứu từ điển và soạn thảo văn bản tiếng Anh” cho phần Nguyên lý hệ điều hành và đề tài “Xây dựng ứng dụng di động đồng bộ ảnh lên server” cho phần Lập trình mạng.

Do thời gian và kiến thức của em còn hạn chế nên không tránh khỏi những sai sót nhất định trong quá trình thực hiện đồ án. Xin cảm ơn sự hỗ trợ từ thầy Nguyễn Tấn Khôi và rất mong nhận được sự góp ý từ quý thầy cô.

**2. Bố cục của đồ án**

Đồ án bao gồm 2 phần, mỗi phần bao gồm các nội dung sau:

*Chương 1: trình bày cơ sở lý thuyết*

*Chương 2: trình bày quá trình phân tích và thiết kế hệ thống, thiết kế chương trình*

*Chương 3: trình bày kết quả thực thi chương trình, đánh giá kết quả và hướng phát triển*

**PHẦN 1: LẬP TRÌNH MẠNG**

**XÂY DỰNG ỨNG DỤNG DI ĐỘNG ĐỒNG BỘ ẢNH LÊN SERVER**

**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

## **1.1. Bộ giao thức TCP/IP**

## **1.1.1. Khái niệm**

TCP/IP là viết tắt của cụm từ Transmission Control Protocol/Internet Protocol. TCP/IP là tập hợp các giao thức (protocol) điều khiển truyền thông giữa tất cả các máy trên Internet. Cụ thể hơn, TCP/IP chỉ rõ cách thức đóng gói thông tin (hay còn gọi là gói tin), được gửi và nhận các máy tính có kết nối với nhau. TCP/IP được phát triển năm 1978 bởi Bob Kahn và Vint Ceft. Ngày nay TCP/IP được sử dụng rộng rãi trong mạng cục bộ cũng như mạng toàn cầu.

## **1.1.2. Các tầng**

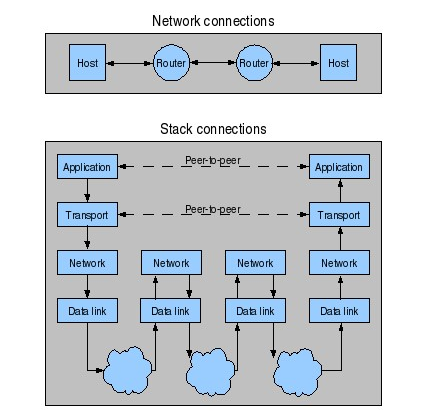
TCP/IP được xem như giản lược của mô hình tham chiếu OSI với 4 tầng ứng dụng như sau:

• Tầng liên kết (Datalink Layer)

• Tầng mạng (Internet Layer)

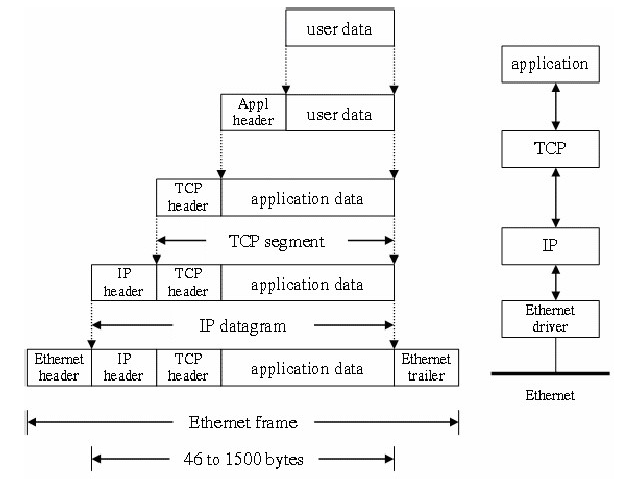
• Tầng giao vận (Transport Layer)

• Tầng ứng dụng (Application Layer)



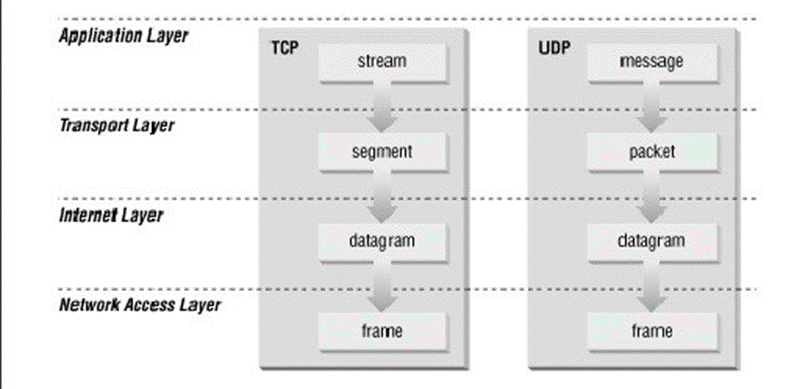
*Hình 1: Sơ đồ bộ giao thức TCP/IP*

## **1.1.3. Phương thức hoạt động**



*Hình 2: Phương thức hoạt động của giao thức TCP/IP*

Cũng tương tự như mô hình OSI, Khi truyền dữ liệu, quá trình tiến hành từ từng trên xuông tầng dưới, qua mỗi tầng dữ liệu được thêm vào thông tin điều khiển gọi là header. Khi nhận dữ liệu thì quá trình xảy ra ngược lại. Dữ liệu được truyền từ tầng dưới lên và qua mỗi tầng thì phần header tâng ứng sẽ được lấy và khi đến tầng trên cùng thì dữ liệu không còn phần header nữa.



*Hình 3: Mô hình các tầng mạng TCP/IP*

Hình trên cho ta thấy được lược đồ dữ liệu qua các tầng. Trong hình ta thấy dữ liệu các tầng khác nhau dữ liệu được mang những thuật ngữ khác nhau

• Trong tầng ứng dụng: dữ liệu là các luồng được gọi là stream

• Trong tầng giao vận: đơn vị dữ liệu mà TCP gửi xuống gọi là TCPsegment

• Trong tầng mạng, dữ liệu mà IP gửi xuống tầng dưới gọi là IP Datagram

• Trong tầng liên kết, dữ liệu được truyền đi gọi là frame

## **1.1.4. Socket và cổng**

Socket: có thể gọi là một ổ kết nối hay một điểm kết nối cho các ứng dụng thông tin với nhau. Đây là một khái niệm trừu tượng (không vật lý) trong lý thuyết mạng, thuộc về phần mềm. Một điểm kết nối (socket) là một kết hợp giữa địa chỉ IP và một cổng kết nối (port). Với khái niệm này, các ứng dụng có thể thông tin với nhau trên cùng một máy tính, hoặc là trên các máy tính khác nhau trên mạng LAN, mạng internet… Một khái niệm không thể thiếu khi sử dụng socket đó là giao thức (protocol), các ứng dụng thông thường trao đổi thông tin, dữ liệu qua giao thức

Cổng (Port): có thể gọi là một cổng kết nối cho mạng thông tin, mạng internet. Cổng kết nối có thể là một cổng thực, vật lý. Tuy nhiên trong lý thuyết mạng cổng kết nối có thể là khái niệm trừu tượng, đánh số từ 0 đến 65535, được dùng trong các lớp phần mềm (không vật lý). Để tránh nhầm lẫn với nhau, mỗi ứng dụng phải dùng riêng ổ kết nối (socket), dùng riêng giao thức (protocol).

## **Giao thức HTTP**

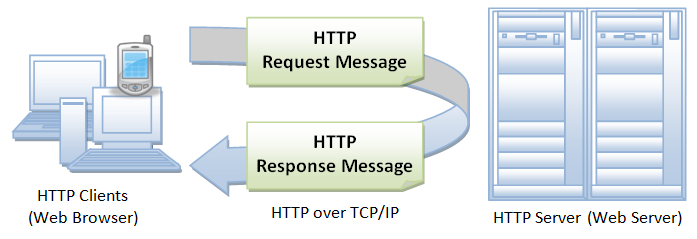
* + 1. **Khái niệm**

Hyper Text Transfer Protocol (HTTP) là một giao thức thuộc tầng ứng dụng được dùng để truyền tải các tài liệu đa phương tiện, ví dụ như HTML. Giao thức này được thiết kế để truyền thông giữa các trình duyệt web và máy chủ web, tuy nhiên nó cũng được dùng cho nhiều mục đích khác. Giao thức HTTP quy định cấu trúc của các gói tin và cách thức truyền nhận dữ liệu giữa client và server thông qua môi trường internet. HTTP tuân theo một mô hình client-server truyền thống, với một client mở một kết nối (connection) để tạo ra một yêu cầu (request), sau đó chờ đợi cho đến khi nó nhận được phản một phản hồi (response). HTTP là một giao thức không lưu lại trạng thái (stateless protocol), có nghĩa là máy chủ không lưu giữ bất cữ dữ liệu (state) gì giữa các yêu cầu. Bởi thường được dựa trên một lớp TCP/IP, nó có thể được sử dụng trên bất cứ tầng giao vận đáng tin cậy nào (reliable transport layer) - những giao thức không bị mất dữ liệu, như là UDP. Với khả năng truyền dẫn siêu văn bản (text, hình ảnh, âm thanh, video,…), HTTP hiện là nền tảng truyền dẫn dữ liệu của ứng dụng duyệt web hiện nay và được ứng dụng rất nhiều trong các hệ thống Internet of Things.

* + 1. **Mô hình**

Quá trình của một phiên làm việc HTTP diễn ra như sau:

1. Client thiết lập một kết nối đến server. Nếu thiết lập thành công, client và server sẽ truyền nhận dữ liệu với nhau thông qua kết nối này, kết nối được thiết lập còn gọi là socket interface bao gồm các thông tin: địa chỉ IP và port.
2. Sau khi kết nối thành công, client gửi HTTP request đến server thông qua socket interface vừa được thiết lập. Trong gói tin request sẽ chứa đường dẫn và phương thức.
3. Server sẽ nhận và xử lý request từ client sau đó đóng gói dữ liệu tương ứng và gửi một HTTP response về cho client. Dữ liệu trả về sẽ là JSON hoặc file HTML chứa các loại dữ liệu khác nhau.
4. Đóng kết nối.



*Hình 4: Mô hình hoạt động của giao thức HTTP*

* + 1. **Định dạng gói tin**

Trong suốt phiên làm việc HTTP như ở trên, có hai loại gói tin chính được trao đổi là gói tin yêu cầu gửi từ client đến server gọi là “Request message” và gói tin phản hồi gửi từ server đến client gọi là “Response message”. Hai loại gói tin này được giao thức HTTP định nghĩa rất cụ thể:

* + - 1. **HTTP request message**

Gồm 3 phần chính là: Request line, Header và Body. Dữ liệu được chia thành các dòng và định dạng kết thúc dòng là <CR><LF> (tương ứng với các ký tự 0x0A,0x0D trong bảng mã ASCII). Riêng dòng cuối cùng của gói tin sẽ được báo hiệu bằng 2 lần định dạng kết thúc: <CR><LF><CR><LF>.

Request line là dòng đầu tiên của gói HTTP Request, bao gồm 3 trường: phương thức (method), đường dẫn (path – có nhiều bài viết gọi là URL hoặc URI cho trường này) và phiên bản giao thức (HTTP version).

* Phương thức (method) có thể là: GET, POST, HEAD và DELETE
* Đường dẫn (path) dùng để định danh nguồn tài nguyên mà client yêu cầu, bắt buộc phải có ít nhất là dấu “/”.
* Phiên bản giao thức (HTTP version): là phiên bản HTTP client đang sử dụng (thường là HTTP/1.0 hoặc HTTP/1.1)
* Header (không bắt buộc) viết ở định dạng “Name:Value” cho phép client gửi thêm các thông tin bổ sung về thông điệp HTTP request và thông tin về chính client.
* Body là dữ liệu gửi từ client đến server trong gói tin HTTP request.
  + - 1. **HTTP response message**

Định dạng gói tin HTTP response cũng gồm 3 phần chính là: Status line, Header và Body.

Status line: gồm 3 trường là phiên bản giao thức (HTTP version), mã trạng thái (Status code) và mô tả trạng thái (Status text):

* Phiên bản giao thức (HTTP version): phiên bản của giao thức HTTP mà server hỗ trợ, thường là HTTP/1.0 hoặc HTTP/1.1
* Mã trạng thái (Status code): mô tả trạng thái kết nối dưới dạng số, mỗi trạng thái sẽ được biểu thị bởi một số nguyên. Ví dụ: 200, 404, 302,…
* Mô tả trạng thái (Status text): mô tả trạng thái kết nối dưới dạng văn bản một cách ngắn gọn, giúp người dùng dễ hiểu hơn so với mã trạng thái. Ví du: 200 OK, 404 Not Found, 403 Forbiden,…

Header line: có chức năng tương tự như gói tin request, giúp server có thể truyền thêm các thông tin bổ sung đến client dưới dạng các cặp “Name:Value”.

Body: là nơi đóng gói dữ liệu để trả về cho client, ở phạm vi đồ án này body là dữ liệu trả về dưới dạng JSON.

* + 1. **Phương thức**

HTTP cung cấp các phương thức:

* **GET** : được sử dụng để lấy lại thông tin từ Server đã cung cấp bởi sử dụng một URI đã cung cấp. Các yêu cầu sử dụng GET chỉ nhận dữ liệu và không có ảnh hưởng gì tới dữ liệu.
* HEAD : Tương tự như GET, nhưng nó truyền tải dòng trạng thái và khu vực Header.
* **POST** : Một yêu cầu POST được sử dụng để gửi dữ liệu tới Server.
* PUT : Thay đổi tất cả các đại diện hiện tại của nguồn mục tiêu với nội dung được tải lên.
* **DELETE** : Gỡ bỏ tất cả các đại diện hiện tại của nguồn mục tiêu bởi URI.
* CONNECT : Thiết lập một tunnel tới Server được xác định bởi URI đã cung cấp.
* OPTIONS : Miêu tả các chức năng giao tiếp cho nguồn mục tiêu.
* TRACE : Trình bày một vòng lặp kiểm tra thông báo song song với path tới nguồn mục tiêu.
  1. **Application Programming Interface**

## **Khái niệm**

Application Programming Interface (giao diện lập trình ứng dụng) viết tắt là API, là phương thức kết nối với các thư viện và ứng dụng khác. Web API là một phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được Web API trả lại thường ở dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS.

## **Phương thức hoạt động**

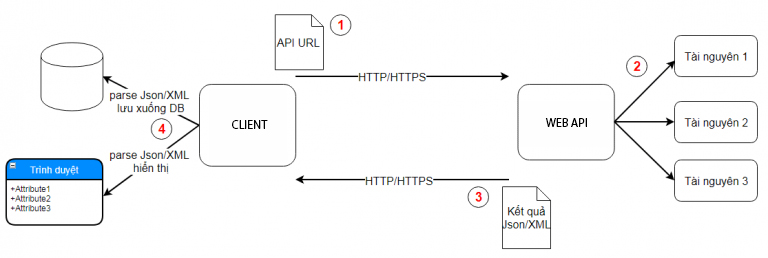
API hoạt động thông qua 4 bước:

Bước 1: Client xây dựng API Connection. Công đoạn này gửi đến máy chủ cung cấp nội dung, dịch vụ thông qua giao thức quen thuộc HTTP hoặc HTTPS.

Bước 2: Tại máy chủ, dịch vụ sẽ thực hiện kiểm tra xác thức (nếu có) và tìm đến nguồn tài nguyên thích hợp để tạo nội dung trả về cho phù hợp.

Bước 3: Máy chủ trả về kết quả dưới định dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS.

Bước 4: Tại client dữ liệu trả về sẽ được phân tích để thực hiện tiếp các hoạt động như lưu dữ liệu về cơ sở dữ liệu, hiển thị dữ liệu,…



*Hình 5: Mô hình hoạt động Client-Server sử dụng API*

* + 1. **Kiến trúc REST**

REST (REpresentational State Transfer) được đưa ra vào năm 2000, trong luận văn tiến sĩ của Roy Thomas Fielding (đồng sáng lập giao thức HTTP). Nó là một dạng chuyển đổi cấu trúc dữ liệu, là một phong cách kiến ​​trúc cho việc thiết kế các ứng dụng có kết nối. Nó sử dụng HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE đến một URL để xử lý dữ liệu.

REST API là một ứng dụng chuyển đổi cấu trúc dữ liệu có các phương thức để kết nối với các thư viện và ứng dụng khác. REST API không được xem là một công nghệ, nó là một giải pháp để tạo ra các ứng dụng services thay thế cho các kiểu khác như SOAP, WSDL (Web Service Definition Language),…

Các ưu điểm của REST

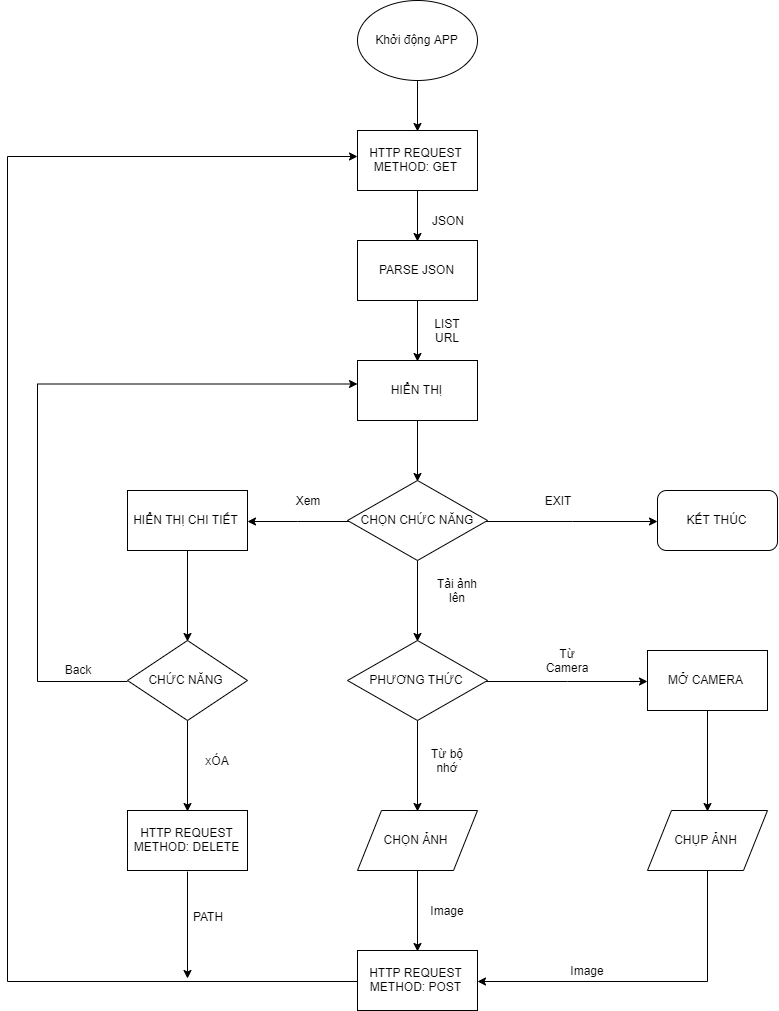
* Giúp cho ứng dụng trở nên rõ ràng hơn.
* REST URL đại diện cho resource chứ không phải là hành động.
* Dữ liệu được trả về với nhiều định dạng khác nhau như: XML, HTML, JSON
* Code đơn giản và ngắn gọn.
* REST chú trọng vào tài nguyên hệ thống.

## **JSON**

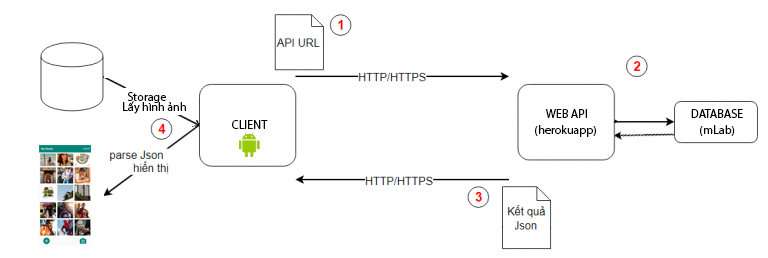
JSON (Javascript Object Notation) là một dạng dữ liệu tuân theo một quy luật nhất định mà hầu hết các ngôn ngữ lập trình hiện nay đều có thể đọc được. JSON có định dạng đơn giản, dễ dàng sử dụng và truy vấn. Cú pháp của JSON rất đơn giản đó là mỗi thông tin dữ liệu sẽ có 2 phần : key và value, điều này tương ứng trong cơ sở dữ liệu là tên field và giá trị của nó ở một record nào đó.

Trong phạm vi đồ án này, kết quả trả về khi thực hiện một request tới server thông qua giao thức HTTP là một JSON. Hiện nay có rất nhiều thư viện hỗ trợ việc xử lý chuỗi JSON trả về. Đó là một lợi thế cực lớn cho việc phát triển JSON về sau.

## **1.4. Thuật toán chương trình**



*Hình 6: Thuật toán làm việc của client*



*Hình 7: Cách thức hoạt động của chương trình*

## **1.5. Kết chương**

Chương này trình bày hai cơ sở nền tảng quan trọng gồm lý thuyết và thuật toán. Từ đó giúp bước đầu định hướng đề tài và tạo cơ sở cho việc phân tích và thiết kế hệ thống

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

## **2.1. Các chức năng**

Client

* Xem danh sách ảnh đã đồng bộ lên server dưới dạng gridview
* Xem chi tiết ảnh đã đồng bộ.
* Xóa ảnh đã đồng bộ.
* Đồng bộ ảnh mới.

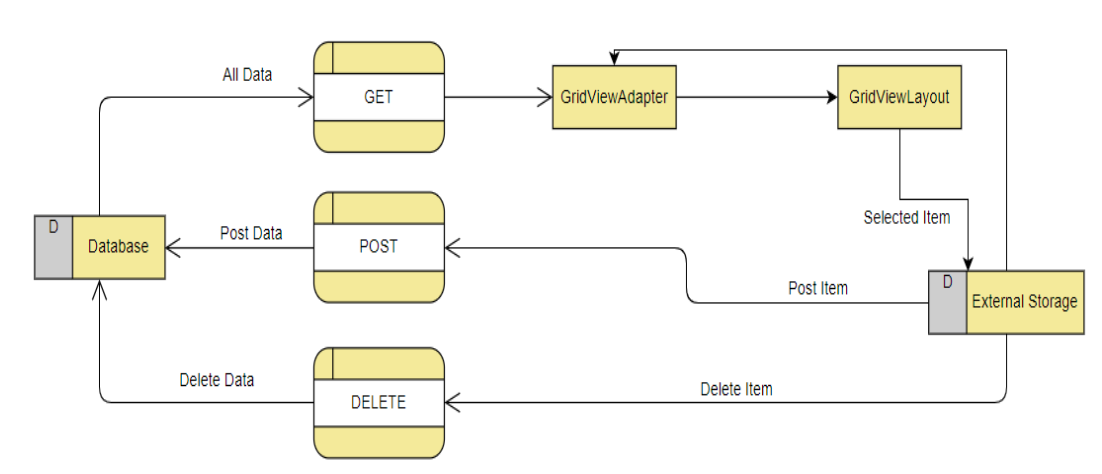
Server

* Server sẽ chờ nhận những lệnh yêu cầu xử lý từ client và trả về kết quả cho client

**2.2. Thiết kế chương trình**

Chương trình bao gồm ứng dụng Android PhotoApp, lập trình sử dụng ngôn ngữ Java và công cụ Android Studio : đóng vai trò là client; server chạy trên PC trong môi trường Node hoặc có thể deploy thành web API trên môi trường internet, lập trình sử dụng ngôn ngữ NodeJS và công cụ Visual Studio Code, sử dụng Postman là công cụ kiểm tra: đóng vai trò là server

## **2.3. Sơ đồ luồng dữ liệu**



*Hình 8: Sơ đồ luồng dữ liệu của ứng dụng đồng bộ ảnh*

## **2.4. Kết chương**

Chương này trình bày các vấn đề về phân tích thiết kế hệ thống bao gồm: thiết kế các chức năng, chương trình, sơ đồ luồng dữ liệu. Qua đó giúp ta xác định rõ ràng mục tiêu, ngôn ngữ lập trình, môi trường, công cụ hỗ trợ để xây dựng chương trình.

**CHƯƠNG 3: TRIỂN KHAI**

## **3.1. Môi trường cài đặt**

## **Ngôn ngữ**

* Java: là một trong những ngôn ngữ lập trình hướng đối tượng phổ biến nhất thế giới.
* NodeJS: là một nền tảng chạy trên môi trường V8 JavaScript runtime - một trình thông dịch JavaScript cực nhanh chạy trên trình duyệt Chrome

## **Môi trường**

* Android: hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng.
* NodeJS: được sử dung để thông dịch và thực thi các đoạn code Javascript trên server, tạo một môi trường ảo trong localhost

## **IDE**

* Android Studio: xây dựng mã nguồn client
* Visual Code: xây dựng mã nguồn server

## **Thư viện hỗ trợ**

* Glide: thư viện load ảnh của tác giả Sam Judd. Dùng để tối ưu hóa công việc load ảnh, tạo thumbnail và hiển thị ảnh từ URL. <https://github.com/bumptech/glide>
* Retrofit: thư viện kết nối API và xử lý JSON trả về từ API. Là một thư viện cực kì phổ biến trong lập trình android.

<https://square.github.io/retrofit/>

* 1. **Kết quả thực thi chương trình**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Hình 9: Xem toàn bộ ảnh* | *Hình 10: Xem chi tiết* | *Hình 11:Chức năng đồng bộ ảnh* |
|  |  |  |
|  |  | |
| *Hình 12: Chức năng xóa ảnh* | *Hình 13: Giao diện Command Prompt của Server* | |

## **3.3. Đánh giá**

## **Ưu điểm**

* Chương trình về cơ bản đã thực hiện được một số phương thức cơ bản của giao thức HTTP như GET, POST, DELETE.
* Chương trình đã xây dựng được giao diện cơ bản để người dùng có thể giao tiếp dễ dàng với server
* Ứng dụng phản hồi nhanh, ổn định.
* Đáp ứng tốt các yêu cầu đã đề ra

## **Nhược điểm**

* Đáp ứng nhu cầu người dùng còn thấp, khả năng quản lý chưa thật sự tối ưu
* Chưa đồng bộ được nhiều ảnh cùng lúc
* Chưa có các chức năng tương tự đối với video
* Chụp ảnh ở camera cho chất lượng kém
* Phải cài đặt API URL thủ công
* Chưa có chức năng xác thực để truy cập tài nguyên

## **Giải pháp đề xuất**

Trong thời gian tìm hiểu, nghiên cứu cơ sở lý thuyết và triển khai ứng dụng, em đã có cơ hội tìm hiểu về họ giao thức TCP/IP nói chung cũng như giao thức HTTP/HTTPS nói riêng, và ứng dụng để xây dựng chương đồng bộ ảnh giữa điện thoại (thông qua server) và máy tính sử dụng ngôn ngữ lập trình Java.

Trong quy mô của đề tài, sản phẩm được triển khai còn khá đơn giản về giao diện, cũng như các chức năng mở rộng và phức tạp chưa được nghiên cứu đến. Một số hướng phát triển của đề tài trong tương lai:

* Xây dựng giao diện ứng dụng bắt mắt và trực quan hơn
* Có thể upload nhiều file trong cùng một lúc
* Cài đặt thêm các lệnh đang thử nghiệm của giao thức
* Nghiên cứu về khía cạnh bảo mật thông tin khi sử dụng
* Có các tính năng tương tự với video
* Tìm hiểu kĩ thuật lấy URI ảnh sau khi lấy nội dung từ camera

## **3.4. Kết chương**

Chương này trình bày về môi trường triển khai ứng dụng, các công cụ hỗ trợ, kết quả đã đạt được cũng như đề xuất giải pháp. Từ đó tạo cơ sở để phát triển ứng dụng sau này.

**PHẦN 2: HỆ ĐIỀU HÀNH**

**TÌM HIỂU KĨ THUẬT LẬP TRÌNH HOOK**

**XÂY DỰNG CHƯƠNG TRÌNH HỖ TRỢ TRA TỪ ĐIỂN TIẾNG ANH**

**CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI**

## **1.1. Cơ sở lý thuyết**

## **Khái niệm về Hook**

Hook là kỹ thuật điều khiến thông điệp (message-handling) giúp cho ứng dụng có thể cài đặt một thủ tục đế điều khiến luồng thông điệp và tiến hành xử lý các thông điệp đó truớc khi thông điệp đó đi tới cửa số đích. Hàm dùng để chặn một loại sự kiện riêng biệt được gọi là hook procedure/ Hook function. Mỗi khi nhận được sự kiện, hook procedure có thể thay đổi và thậm chí hủy bỏ các sự kiện đó.

Các mô hình Hook:

- Local hook: là kỹ thuật Hook dùng để bẫy sự kiện ngay trong tiến trình cài đặt.

- Remote hook: là kỹ thuật Hook cho phép bẫy các sự kiện thuộc tiến trình của ứng dụng khác. Trong mô hình này lại tồn tại hai kiểu hook khác :

+ Thread-specific : kiểu Hook này sẽ bẫy sự kiện của một luồng cụ thể.

+ System-wide : bẫy sự kiện của tất cả các luồng trong tất cả các tiến trình đang thi hành trong hệ thống.

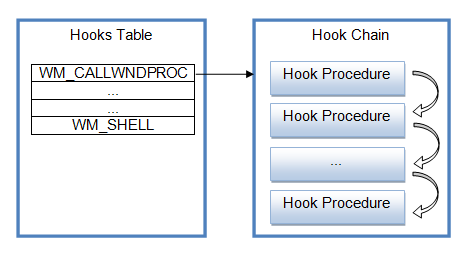
## **Cơ chế**

Có nhiều loại hook (như chuột, bàn phím) và hệ điều hành luôn duy trì một danh sách các hook procedure cho mỗi loại đó. Mỗi danh sách các Hook procedure này được gọi là hookchain(chuỗi hook) . Bản chất của hook chain là một dãy các con trỏ hàm trỏ đến các Hook procedure.

Khi hệ thống thực hiện một sự kiện nào đó, nó sẽ tìm kiếm trong hook chain tương ứng với sự kiện đó. Nếu một hook procedure phù hợp được tìm thấy, hệ thống sẽ thực hiện nó và chỉ lấy lại quyền điều khiển sau khi hook chain kết thúc. Vì thế khi một hook procedure thực hiện xong, nó phải thực hiện việc chuyển quyền điều khiển cho hook procedure kế tiếp trong hook chain.

Tuy nhiên cơ chế này còn tùy thuộc vào loại hook. Như một số loại hook chỉ có thể theo dõi các thông điệp, vì vậy cho dù hook procedure có chuyển quyền điều khiển cho hook procedure kế tiếp hay không, hệ thống vẫn sẽ tự động làm việc này.

Một điểm cần lưu ý là hook sẽ làm chậm hệ thống, vì thế chỉ nên cài đặt hook khi cần thiết và loại bỏ nó khi đã hoàn tất công việc.



*Hình 14: Cơ chế của Hook*

Hook có thế xem là tính năng mạnh mẽ nhất của Windows, nó cho phép ta đặt bẫy đối với các sự kiện. Bằng cách sử dụng hook, ta có thế điều huớng ứng dụng tới một thủ tục mới bất kế khi nào mà sự kiện đuợc quan tâm xuất hiện và bất kế sự kiện đó thuộc tiến trình của bạn hay thuộc tiến trình khác.

Để viết đuợc chuơng trình Hook ta cần nắm bắt cơ chế xử lý thông điệp của Windows, thư viện Win32 API và kỹ thuật Hook.

* + 1. **Các loại Hook**

Có nhiều loại hook được phân biệt dựa vào các sự kiện, thông điệp mà Hook procedure can thiệp vào.

|  |  |
| --- | --- |
| WH\_CALLWNDPROC WH\_CALLWNDPROCRET | Cho phép theo dõi các thông điệp gởi đến các thủ tục cửa sổ (window procedure) |
| WH\_CBT | Gọi trước khi activating, creating, destroying, minimizing, maximizing, moving, hoặc sizing một cửa sổ ; trước khi hoàn thành một lệnh hệ thống, trước khi loại bỏ một sự kiện chuột hoặc bàn phím từ hàng đợi thông điệp hệ thống … |
| WH\_DEBUG | Gọi trước khi gọi một thủ tục hook kết hợp với bất kỳ hook khác trong hệ thống |
| WH\_FOREGROUNDIDLE | Cho phép bạn thực hiện các tác vụ ưu tiên thấp trong thời gian tác vụ đó đang ở chế độ foreground idle |
| WH\_GETMESSAGE | Cho phép một ứng dụng theo dõi các thông điệp trả về bởi hàm GetMessage hoặc PeekMessage |
| WH\_JOURNALPLAYBACK | Cho phép một ứng dụng chèn thêm các thông điệp vào hàng đợi thông điệp hệ thống. |
| WH\_JOURNALRECORD | Cho phép bạn giám sát và ghi lại các sự kiện đầu vào. |
| WH\_KEYBOARD\_LL | Cho phép bạn theo dõi các sự kiện bàn phím trong hàng đợi ứng dụng |
| WH\_KEYBOARD | Cho phép một ứng dụng theo dõi lưu lượng truy cập thông điệp của thông điệp WM\_KEYDOWN và WM\_KEYUP được trả về bởi hàm GetMessage hoặc PeekMessage |
| WH\_MOUSE\_LL | Cho phép bạn theo dõi các sự kiện chuột trong hàng đợi ứng dụng |
| WH\_MOUSE | Cho phép bạn theo dõi các thông điệp chuột được trả về bởi hàm GetMessage hoặc PeekMessage |
| WH\_MSGFILTER WH\_SYSMSGFILTER | Cho phép theo dõi chính các thông điệp được xử lý bởi menu, scrollbar, dialog… |
| WH\_SHELL |  |

* + 1. **Khả năng**

Hook cung cấp các khả năng mạnh cho các ứng dụng chạy trên nền Windows, các ứng dụng này có thể dùng hook để :

* Xử lý hoặc định nghĩa tất cả các thông điệp cho dialog box, message box, scroll bar, hoặc menu của một ứng dụng. (Sử dụng hook WH\_MSGFILTER).
* Xử lý hoặc định nghĩa tất cả các thông điệp cho dialog box, message box, scroll bar, hoặc menu của hệ thống. (Sử dụng hook WH\_SYSMSGFILTER).
* Xử lý hoặc định nghĩa tất cả các thông điệp (bất chấp là thông điệp gì) của hệ thống mỗi khi GetMessage hoặc PeekMessage được gọi. (Sử dụng hook WH\_GETMESSAGE)
* Xử lý hoặc định nghĩa tất cả các thông điệp (bất chấp là thông điệp gì) của hệ thống mỗi khi SendMessage được gọi. (Sử dụng hook WH\_CALLWNDPROC).
* Thu (Record) và phát lại (Playback) các sự kiện keyboard và mouse . (Sử dụng hook WH\_JOURNALRECORD, WH\_JOURNALPLAYBACK).
* Xử lý , định nghĩa hoặc hủy bỏ tất cả các sự kiện bàn phím.(Sử dụng hook WH\_KEYBOARD).
* Xử lý , định nghĩa hoặc hủy bỏ tất cả các sự kiện chuột (Sử dụng hook WH\_MOUSE).
  + 1. **Ứng dụng**

Tận dụng các khả năng trên, các ứng dụng có thể sử dụng hook để :

* Cung cấp phím trợ giúp F1 hỗ trợ menu, dialog box và message box (Sử dụng hook WH\_MSGFILTER).
* Cung cấp các tính năng thu và phát các sự kiện mouse và keyboard, thường được gọi là các macro. (Sử dụng hook WH\_JOURNALRECORD, WH\_JOURNALPLAYBACK).
* Theo dõi các thông điệp để biết được thông điệp nào được gởi đến cửa sổ nào cũng như các hành động nào sẽ làm phát sinh thông điệp tương ứng. (Sử dụng hook WH\_GETMESSAGE & WH\_CALLWNDPROC). Chương trình Spy trong bộ Win32™ SDK của Windows NT đã rất thành công trong việc sử dụng hook để thực hiện tác vụ này.
* Giả lập các tác vụ input của keyboard và mouse (Sử dụng hook WH\_JOURNALPLAYBACK). Chỉ có hook mới có thể cho ta một phương pháp chắc chắn và tin cậy để thực hiện điều này. Nếu ta tiếp cận theo cách khác, gởi message chẳng hạn, Windows sẽ không cập nhật trạng thái của mouse cũng như keyboard, và điều này sẽ dẫn đến những hành động không mong muốn. Còn nếu như hook được sử dụng, các sự kiện sẽ được xử lý y hệt như sự kiện vật lý.

Một số ứng dụng điển hình của Hook như sau:

– Tạo các chương trình record và play back macro.

– Tạo các ứng dụng computer-based training (CBT)

– Bắt và giả lập các thông điệp bàn phím, chuột

– Cung cấp chức năng Help (F1) cho ứng dụng

– Xác định trạng thái rỗi của ứng dụng để thực hiện công việc nào đó.

– Tạo các chức năng debug.

* + 1. **Phạm vi của Hook**

Hook có 2 loại: hook procedure có phạm vi cục bộ (local hook/thread hook) hoặc toàn cục (global hook). Tất cả các loại hook đểu có thể được cài đặt để trở thành global hook, một số cho phép chọn lựa phạm vi hook dựa vào các tham số trong hàm cài đặt (SetWindowsHookEx).

– Local hook (thread hook): chỉ có ảnh hưởng trong phạm vi một thread.

– Global hook: có ảnh hưởng trong toàn hệ thống. Trường hợp này, hook procedure phải được chứa trong một thư viện DLL.

* + 1. **Các hàm Win32 cần thiết**

Để sử dụng hook, ta cần sử dụng ba hàm Win32 của Windows là:

– SetWindowsHookEx: cài đặt một hook procedure vào một hook chain.

– CallNextHookEx: chuyển quyền điều khiển cùng các thông tin hook cho hook procedure kế tiếp trong hook chain. Bạn có thể không sử dụng hàm này tuy nhiên chỉ khi nào bạn muốn chặn các hook procedure còn lại trong hook chain.

– UnhookWindowsHookEx: gỡ hook procedure ra khỏi hook chain được cài đặt bởi SetWindowsHookEx.

* + 1. **Cài đặt Hook**

Ta có thể cài đặt thủ tục hook vào chuỗi hook bằng việc gọi hàm SetWindowsHookEx và chỉ ra kiểu hook đang gọi thủ tục, việc cài đặt hook có thể thực hiện trên mọi tiến trình trong hệ thống.

Nếu sử dụng hook toàn cục thì phải đặt trong thư viện liên kết động (DLL). Ứng dụng muốn sử dụng thư viện liên kết động phải lấy được handle của thư viện đó. Để nhận Handle của thư viện liên kết động ta có thể sử dụng hàm LoadLibrary với tham số là tên của thư viện. Sau khi có được Handle của DLL, ta sẽ lấy địa chỉ của thủ tục hook trong thư viện liên kết động thông qua hàm GetProcAddress. Sau khi đã có thủ tục hook, sử dụng hàm SetWindowsHookEx để cài đặt thủ tục hook vào trong chuỗi hook. Ở phạm vi đồ án này, chúng ta sử dụng thư viện **user32.dll** cho việc lập trình với hook.

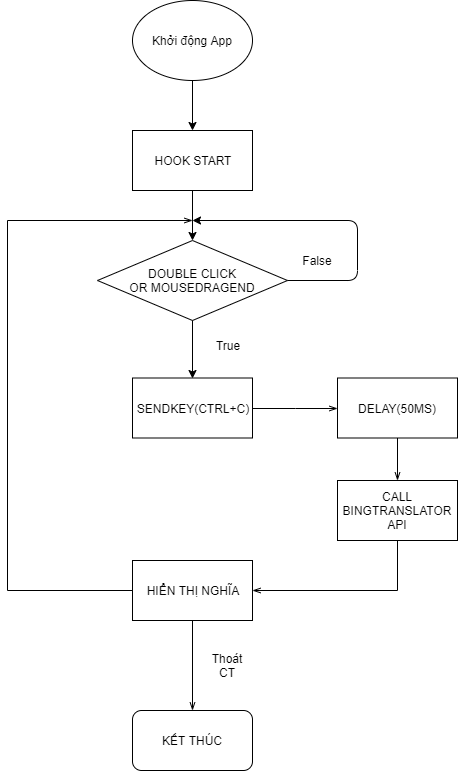
* + 1. **Giải phóng Hook**

Hook nên được bỏ đi nếu như không cần thiết nữa bằng cách sử dụng hàm UnhookWindowsHookEx.

Với thread-specific hook, việc sử dụng hàm UnhookWindowsHookEx sẽ giải phóng thủ tục hook. Tuy nhiên với hook toàn tục (system-wide hook) thì hàm này không thể trả tự do cho hàm DLL. Việc gọi hàm LoadLibrary sẽ gọi trong ngữ cảnh của tất cả các tiến trình, tuy nhiên hàm FreeLibrary thì không thể thực hiện với các tiến trình khác. Vì vậy, không có cách nào để giải phóng DLL. Hệ thống chỉ có thể giải phóng DLL khi tất cả các tiến trình liên kết tới DLL đó phải kết thúc hoặc gọi FreeLibrary.

Giải pháp đặt ra cho vấn đề này là xây dựng hàm cài đặt ngay trong thư viện DLL. Bằng việc liên kết tới DLL, ứng dụng có thể cài đặt hook. Và ngay trong DLL cũng phải có hàm giải phóng hook để giải phóng khi không cần đến nữa.

## **Thuật toán chương trình**



*Hình 15: Sơ đồ thuật toán ứng dụng hỗ trợ tra từ điển*

## **Kết chương**

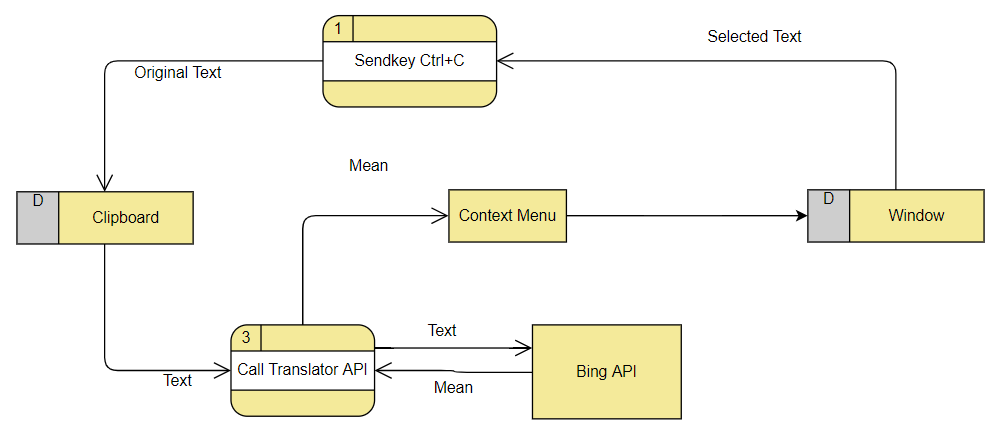
Chương này trình bày cơ sở nền tảng gồm cơ sở lý thuyết về hook, cách cài đặt hook, thuật toán chính. Từ đó giúp bước đầu định hướng đề tài và công việc cần làm, là cơ sở để phân tích và thiết kế hệ thống.

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

## **2.1. Các chức năng**

* Soạn thảo văn bản: chương trình phải chứa một trình soạn thảo văn bản có đầy đủ các tính năng cơ bản: Soạn thảo, lưu file, đọc file, các thao tác: copy, cut, paste, align,…
* Dịch nghĩa: khi chọn bất kì từ nào trong phạm vi window hoặc trong chính trình soạn thảo văn bản của trương trình, khi nhả chuột thì nghĩa sẽ hiển thị ngay kế bên
* Gợi ý từ: ứng dụng sẽ gợi ý những từ tiếng anh có nghĩa tương tự chức năng nhắc lệnh của các editor hiện nay

## **2.2. Sơ đồ luồng dữ liệu**



*Hình 16: Sơ đồ luồng dữ liệu ứng dụng dịch nghĩa*

## **2.3. Kết chương**

Chương này trình bày các vấn đề về phân tích thiết kế hệ thống. Qua đó giúp ta hiểu rõ về các chức năng, sơ đồ luồng dữ liệu khi thực hiện các chức năng. Là tiền đề để tiến hành xây dựng ứng dụng.

**CHƯƠNG 3: KẾT QUẢ THỰC THI**

## **Môi trường thực thi**

## **Ngôn ngữ**

C# là một trong những ngôn ngữ lập trình hướng đối tượng phổ biến và mạnh mẽ và là cách tốt nhất giúp ta tiếp cận các hoạt động của hệ điều hành

## **Nền tảng**

Microsoft Window: hệ điều hành dựa trên giao diện người dùng đồ hoạ được phát triển và được phân phối bởi.

## **IDE**

Visual Studio: là một môi trường phát triển tích hợp từ Microsoft. Nó được sử dụng để phát triển chương trình máy tính cho Microsoft Windows, cũng như các trang web, các ứng dụng web và các dịch vụ web Cung cấp những công cụ và thư viện tốt nhất để triển khai Hook

## **Kết quả thực nghiệm**

**Graphical user interface, text, application

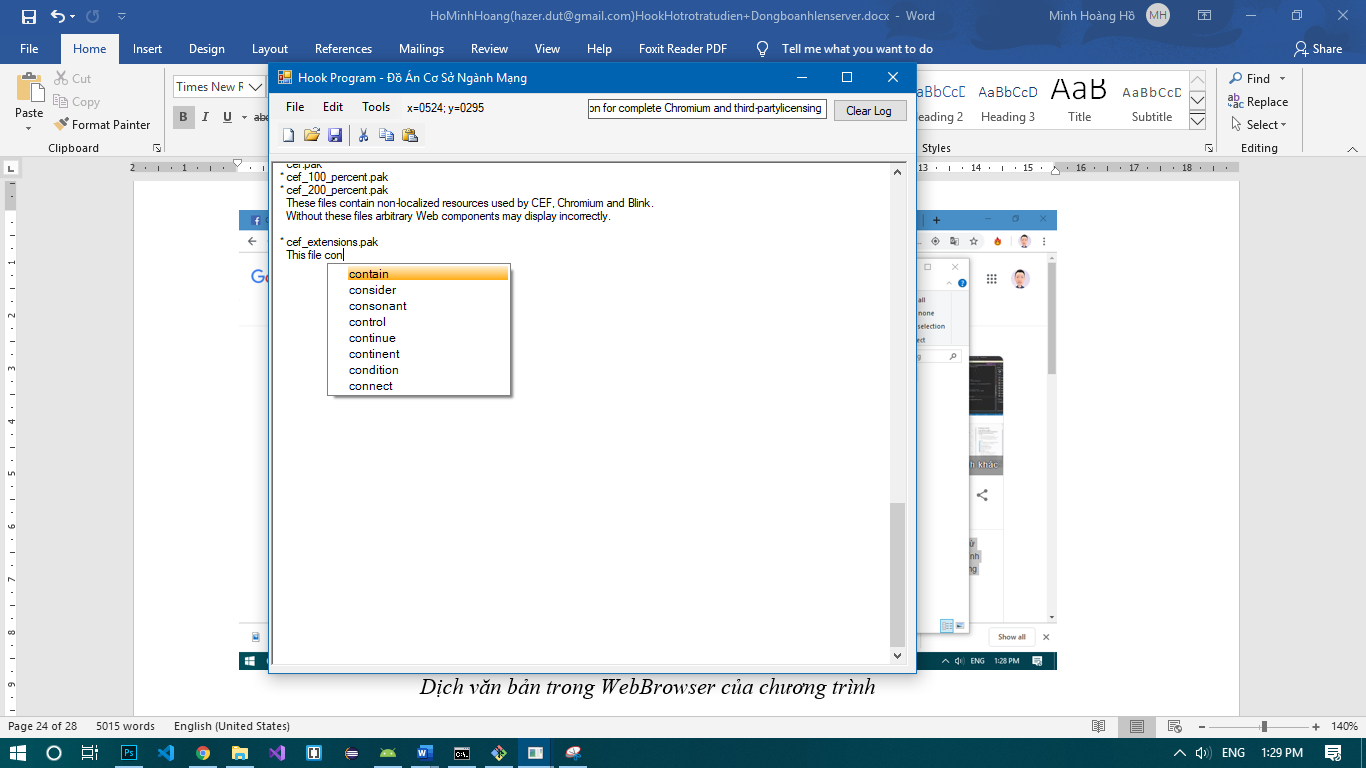
Description automatically generated**

*Hình 17: Giao diện chính của chương trình*

**Graphical user interface, text, application, email

Description automatically generated**

*Hình 18: Soạn thảo và dịch văn bản*



*Hình 19: Gợi ý từ tiếng Anh có nghĩa*

**Graphical user interface, text, application, email

Description automatically generated**

*Hình 20: Dịch văn bản*

## **Đánh giá**

## **Ưu điểm**

* Chương trình về cơ bản đã hoàn thành được yêu cầu của bài toán
* Xây dựng được giao diện cơ bản, thân thiện người dùng

## **Nhược điểm**

* Chức năng dịch bị phụ thuộc vào API của Google
* Chức năng soạn thảo văn bản có ít tính năng

## **Hướng phát triển**

Đề tài này đã giúp em tìm hiểu được kỹ hơn về sự kiện và các vấn đề về sự kiện cũng như cách sử dụng hook để can thiệp nó. Từ đó, em có thể vận dụng để giải quyết một bài toán thực tế đơn giản, xây dựng chương trình hỗ trợ tra cứu từ điển và soạn thảo văn bản tiếng Anh.

Đây chỉ là bài toán demo nên phần xử lý về thuật toán và giao diện vẫn còn đơn giản. Trong tương lai, em sẽ cố gắng tìm hiểu, phát triển sâu hơn về thuật toán để có thể xử lý các bài toán tổng quát và phức tạp hơn, đồng thời nâng cấp về phần giao diện, sử dụng đồ thị hoặc hình ảnh trực quan, hoạt họa để biểu diễn bài toán và quá trình giải quyết vấn đề.

## **Kết chương**

Chương này đã trình bày về môi trường triển khai ứng dụng, các công cụ hỗ trợ, kết quả đã đạt được cũng như đề xuất giải pháp, hướng phát triển. Từ đó tạo cơ sở để phát triển ứng dụng sau này.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. **KẾT QUẢ ĐẠT ĐƯỢC**

Trong thời gian tìm hiểu, nghiên cứu cơ sở lý thuyết và triển khai ứng dụng công nghệ, đồ án đã đạt được những kết quả sau:

*Về mặt lý thuyết,* đồ án đã tìm hiểu được những vấn đề cơ bản về bộ giao thức TCP/IP và giao thức HTTP/HTTPS, tìm hiểu và xây dựng ứng dụng Android, tìm hiểu và xây dựng RestAPI bằng NodeJS, truyền file và nhận JSON thông qua API. Ngoài ra đồ án cũng tìm hiểu được về kĩ thuật lập trình Hook.

*Về mặt thực tiễn ứng dụng*, đồ án đã đạt xây dựng được một ứng dụng hoàn chỉnh chạy trên điện thoại android để đồng bộ ảnh với server. Ứng dụng có đầy đủ chức năng và giao diện thân thiện. Ngoài ra còn xây dựng được chương trình chạy trên nền tảng window hỗ trợ việc soạn thảo văn bản tiếng anh và tra từ điển. Ứng dụng có đầy đủ chức năng và giao diện thân thiện

Tuy nhiên, đồ án còn tồn tại các vấn đề như sau:

* Phụ thuộc vào Bing Translator API
* Chưa có nhiều tính năng
* Chưa xây dựng được đồng bộ với video
* Chưa xử lý thông báo lỗi khi mất kết nối với server
* Chưa xây dựng được thiết đặt API trực tiếp trên ứng dụng, chưa có chức năng đăng nhập

1. **HƯỚNG PHÁT TRIỂN**

Một số số hướng nghiên cứu và phát triển của đề tài như sau:

* Bổ sung thêm các tính năng còn thiếu.
* Sử dụng các thư viện mở rộng để tối ưu hóa đồ án.
* Xây dựng một Translate API của riêng mình.

TÀI LIỆU THAM KHẢO

**Tiếng Việt**

1. Nguyễn Phương Lan & Hoàng Đức Hải (2001), *JAVA – Lập trình mạng*, Nhà xuất bản Lao động Hà Nội.
2. Trần Hồ Thủy Tiên, *Bài giảng Nguyên lý hệ điều hành*, Khoa CNTT Trường Đại học Bách khoa – Đại học Đà Nẵng.

**Tiếng Anh**

1. Andrew S. Tanenbaum & Herbert Bos, *“Modern Operating Systems”*.
2. Elliotte Rusty Harold, *“Java Network Programming”*.

**Internet**

1. [*https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-callnexthookex?redirectedfrom=MSDN*](https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-callnexthookex?redirectedfrom=MSDN)
2. [*https://yinyangit.wordpress.com/2011/08/07/win32-introduction-to-hook-and-some-basic-concepts/*](https://yinyangit.wordpress.com/2011/08/07/win32-introduction-to-hook-and-some-basic-concepts/)
3. [*https://github.com/gmamaladze/globalmousekeyhook*](https://github.com/gmamaladze/globalmousekeyhook)
4. [*https://www.codeproject.com/Articles/365974/Autocomplete-Menu*](https://www.codeproject.com/Articles/365974/Autocomplete-Menu)

PHỤ LỤC:

MÃ NGUỒN CHƯƠNG TRÌNH

1. Mã nguồn lập trình mạng :

Client: <https://github.com/hazerdut/PhotoApp>

Server: <https://github.com/hazerdut/NodeServer>

1. Mã nguồn hệ điều hành: <https://github.com/hazerdut/NetworkOS>