Mô tả source code readMail

-Các thư viện sử dụng:

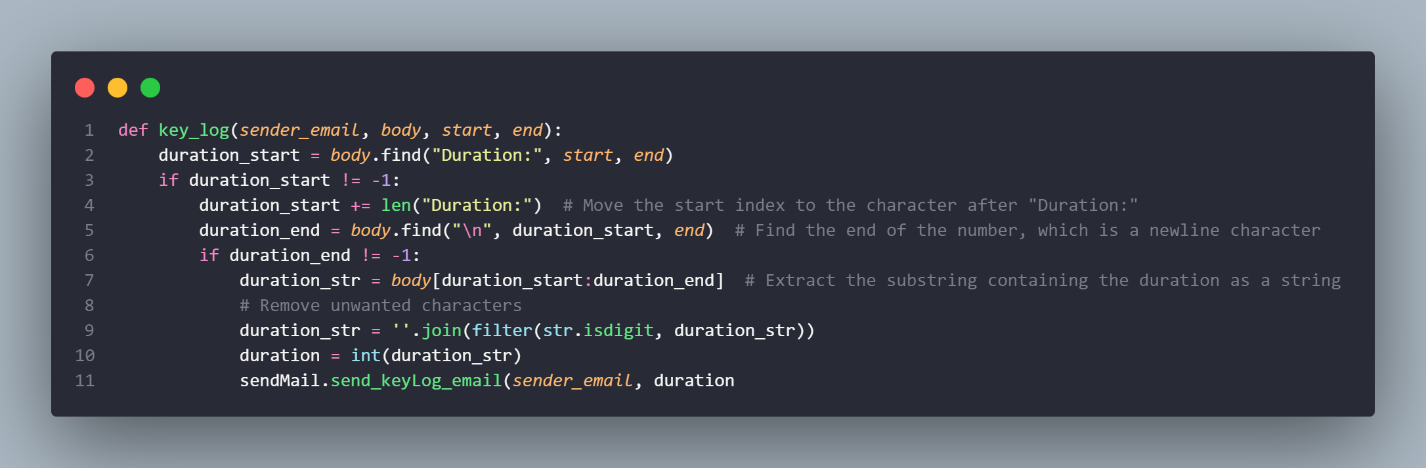
1. **imaplib**: Thư viện hỗ trợ các thao tác với IMAP (Internet Message Access Protocol) để truy cập và quản lý email trên máy chủ.
2. **email**: để xử lý và phân tích email, bao gồm chuyển đổi email từ dạng byte sang đối tượng email và trích xuất thông tin từ email như tiêu đề, người gửi, và nội dung.
3. **email.header**: để giải mã các phần tiêu đề của email.
4. **Timer** và **time**: để đặt và quản lý thời gian và đợi cho các hoạt động cụ thể.
5. **re**: để thực hiện các thao tác với biểu thức chính quy.
6. **BeautifulSoup** từ **bs4**: được sử dụng để để loại bỏ các thẻ HTML từ nội dung email.

-Các hàm đã viết:

+Hàm **extract\_email\_information**

* **email\_message = email.message\_from\_bytes(raw\_email)**: Chuyển đổi dữ liệu email từ dạng byte sang đối tượng email bằng cách sử dụng **email.message\_from\_bytes**. Điều này cho phép ta tương tác với thông tin trong email như tiêu đề, người gửi và nội dung.
* **subject, encoding = decode\_header(email\_message["Subject"])[0]**: Trích xuất thông tin về tiêu đề của email. Hàm **decode\_header** được sử dụng để giải mã tiêu đề vì có thể có mã hóa đặc biệt trong tiêu đề. Nếu tiêu đề là dạng bytes, nó sẽ được giải mã bằng cách sử dụng encoding tương ứng.
* **from\_, encoding = decode\_header(email\_message["From"])[0]**: Trích xuất thông tin về người gửi của email. Tương tự như tiêu đề, thông tin người gửi cũng có thể được mã hóa và cần giải mã nếu là dạng bytes.
* **sender\_email = email\_message["From"]**: Lấy địa chỉ email của người gửi từ email.
* **match = re.search(r'<([^>]+)>', sender\_email)**: Sử dụng biểu thức chính quy để tìm kiếm địa chỉ email trong dấu ngoặc nhọn (**<...>**). Nếu tìm thấy, **sender\_email** sẽ được cập nhật với địa chỉ email được tìm thấy.
* **print("Subject:", subject)**, **print("From:", from\_)**: Hiển thị thông tin về tiêu đề và người gửi email. Đây là dữ liệu được in ra để kiểm tra khi phân tích email.
* **if email\_message.is\_multipart():**: Kiểm tra xem email có nhiều phần không. Nếu có, đoạn mã sẽ lặp qua từng phần để trích xuất nội dung.
* **for part in email\_message.walk():**: Lặp qua từng phần trong email.
* **content\_type = part.get\_content\_type()**: Lấy loại nội dung của phần email hiện tại.
* **content\_disposition = str(part.get("Content-Disposition"))**: Lấy thông tin về việc bố trí nội dung của phần email hiện tại.
* **try: ... except: ...**: Thử giải mã nội dung phần email hiện tại. Nếu không thể giải mã, nó sẽ bỏ qua lỗi và không làm gì cả.
* **if content\_type == "text/plain" and "attachment" not in content\_disposition: ...**: Kiểm tra xem nội dung của phần email hiện tại có phải là văn bản thuần túy không và không phải là file đính kèm. Nếu đúng, in ra nội dung văn bản này.
* **return sender\_email, body**: Trả về địa chỉ email của người gửi và nội dung của email (body) cho việc xử lý tiếp theo.

**Hàm key\_log(sender\_email, body, start, end):**

****

1. **duration\_start = body.find("Duration:", start, end)**: Tìm vị trí xuất hiện đầu tiên của chuỗi "Duration:" trong phần của văn bản được chỉ định.
2. **if duration\_start != -1: ...**: Kiểm tra xem chuỗi "Duration:" có được tìm thấy không.
3. **duration\_start += len("Duration:")**: Di chuyển chỉ số bắt đầu đến ký tự sau "Duration:".
4. **duration\_end = body.find("\n", duration\_start, end)**: Tìm vị trí kết thúc của số liệu, thường là ký tự xuống dòng "\n".
5. **if duration\_end != -1: ...**: Kiểm tra xem kết thúc của số liệu có được tìm thấy không.
6. **duration\_str = body[duration\_start:duration\_end]**: Trích xuất chuỗi chứa thời lượng như một chuỗi ký tự.
7. **duration\_str = ''.join(filter(str.isdigit, duration\_str))**: Loại bỏ các ký tự không phải là số từ chuỗi số liệu.
8. **duration = int(duration\_str)**: Chuyển chuỗi số liệu thành một số nguyên.
9. **sendMail.send\_keyLog\_email(sender\_email, duration)**: Gửi email với thông tin về thời lượng và người gửi email (**sender\_email**).

**Các hàm get\_name\_app\_or\_process(body, stringstart):**

****

Các hàm này thực hiện việc trích xuất tên của ứng dụng hoặc quá trình từ văn bản dựa trên chuỗi bắt đầu được chỉ định (**stringstart**).

1. **start\_index = body.find(stringstart)**: Tìm vị trí xuất hiện đầu tiên của chuỗi bắt đầu được chỉ định (**stringstart**) trong văn bản.
2. **Name = ""**: Khởi tạo biến **Name** rỗng.
3. **if start\_index != -1: ...**: Kiểm tra xem chuỗi bắt đầu được tìm thấy không.
4. **name\_start = start\_index + len(stringstart)**: Xác định vị trí bắt đầu của tên ứng dụng hoặc quá trình.
5. **name\_end = body.find("\n", name\_start)**: Tìm vị trí kết thúc của tên, thường là ký tự xuống dòng "\n".
6. **if name\_end != -1: ...**: Kiểm tra xem kết thúc của tên có được tìm thấy không.
7. **Name = body[name\_start:name\_end].strip()**: Trích xuất tên ứng dụng hoặc quá trình và loại bỏ khoảng trắng xung quanh nó.
8. **Name = BeautifulSoup(Name, "html.parser").text**: Loại bỏ các thẻ HTML từ tên (nếu có).
9. **return Name**: Trả về tên đã được xử lý.

**Các hàm end\_process(sender\_email, body), start\_process(sender\_email, body), start\_app(sender\_email, body), end\_app(sender\_email, body):**



Các hàm này sử dụng hàm **get\_name\_app\_or\_process()** để trích xuất tên ứng dụng hoặc quá trình từ văn bản, sau đó gửi email với thông tin tương ứng nếu tên được tìm thấy.

-Hàm **process\_email**

****

1. **start = 0** và **end = len(body)**: Xác định vị trí bắt đầu và kết thúc trong chuỗi **body**.
2. **if body.find("KeyLog", start, end) != -1: ...**: Kiểm tra xem chuỗi "KeyLog" có xuất hiện trong văn bản không. Nếu có, gọi hàm **key\_log()** để xử lý thông tin về Keylog.
3. **if body.find("ListApp", start, end) != -1: ...**: Tương tự, kiểm tra chuỗi "ListApp" có xuất hiện trong văn bản không. Nếu có, gửi email liệt kê các ứng dụng.
4. **if body.find("Screenshot", start, end) != -1: ...**: Kiểm tra chuỗi "Screenshot" có xuất hiện trong văn bản không. Nếu có, gửi email chứa ảnh chụp màn hình.
5. **if body.find("Shut Down", start, end) != -1: ...**: Kiểm tra chuỗi "Shut Down" có xuất hiện trong văn bản không. Nếu có, thực hiện thao tác tắt máy.
6. **if body.find("Log out", start, end) != -1: ...**: Kiểm tra chuỗi "Log out" có xuất hiện trong văn bản không. Nếu có, thực hiện thao tác đăng xuất.
7. **if body.find("ListProcess", start, end) != -1: ...**: Kiểm tra chuỗi "ListProcess" có xuất hiện trong văn bản không. Nếu có, gửi email liệt kê các quá trình.
8. **if body.find("StartProcess", start, end) != -1: ...**: Kiểm tra chuỗi "StartProcess" có xuất hiện trong văn bản không. Nếu có, gọi hàm **start\_process()** để xử lý thông tin khởi đầu của quá trình.
9. **if body.find("EndProcess", start, end) != -1: ...**: Kiểm tra chuỗi "EndProcess" có xuất hiện trong văn bản không. Nếu có, gọi hàm **end\_process()** để xử lý thông tin kết thúc của quá trình.
10. **if body.find("StartApp", start, end) != -1: ...**: Kiểm tra chuỗi "StartApp" có xuất hiện trong văn bản không. Nếu có, gọi hàm **start\_app()** để xử lý thông tin khởi đầu của ứng dụng.
11. **if body.find("EndApp", start, end) != -1: ...**: Kiểm tra chuỗi "EndApp" có xuất hiện trong văn bản không. Nếu có, gọi hàm **end\_app()** để xử lý thông tin kết thúc của ứng dụng. (Lưu ý: Có một lỗi nhỏ ở đây, có thể bạn muốn sửa thành **end\_app(sender\_email, body)** thay vì **end\_app()(sender\_email, body)**.)

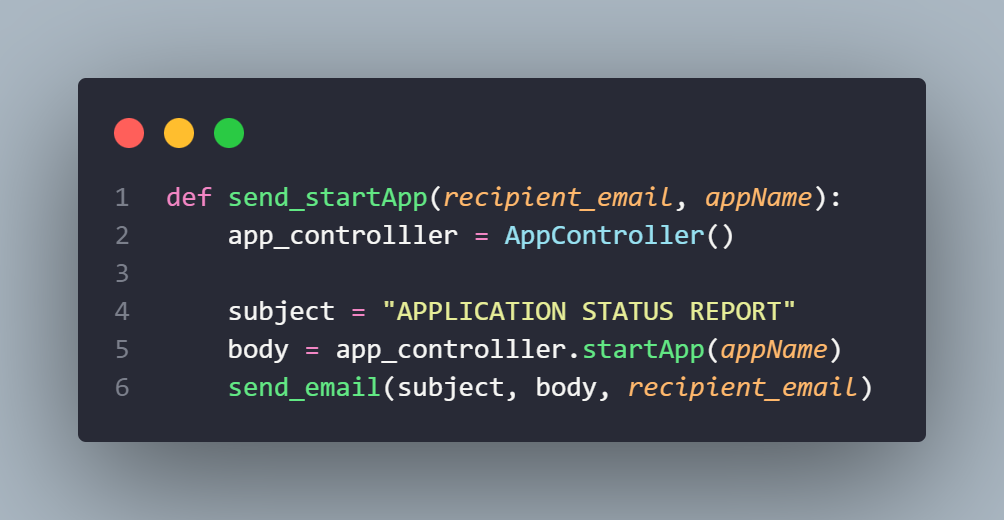
-hàm **read\_email**: thực hiện việc đăng nhập vào tài khoản Gmail, tìm và lấy email chưa đọc, rồi xử lý thông tin từ email cuối cùng được tìm thấy.

Top of Form



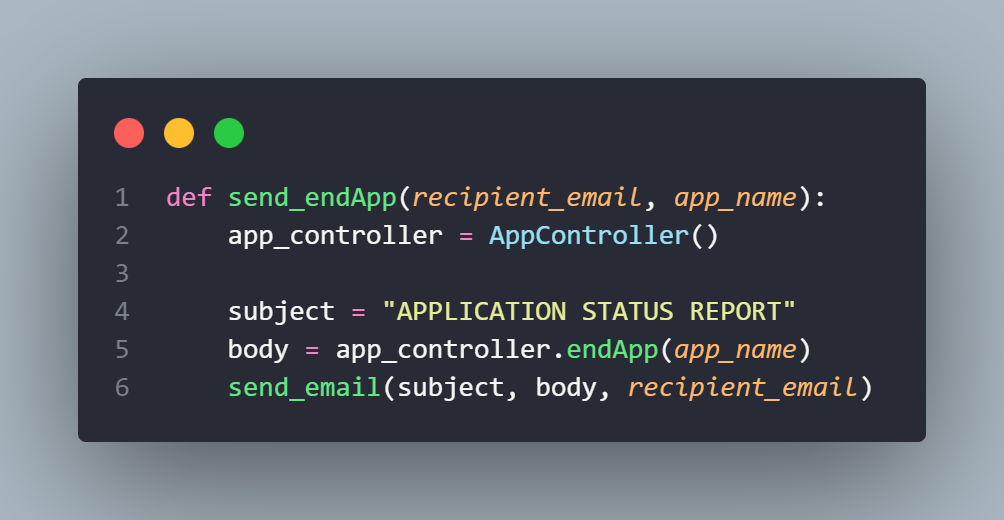
1. **username = "2d2h.computernetwork.clc.fitus@gmail.com" và password = "jvys wjpg mmmx belp"**: Đây là tên người dùng và mật khẩu của tài khoản Gmail được sử dụng để đăng nhập và truy cập email.
2. **imap = imaplib.IMAP4\_SSL("imap.gmail.com")**: Tạo một kết nối SSL với máy chủ IMAP của Gmail để lấy email.
3. **imap.login(username, password)**: Xác thực với tên người dùng và mật khẩu đã cung cấp.
4. **imap.select("inbox")**: Chọn hộp thư "inbox" để thao tác.
5. **status, email\_ids = imap.search(None, 'UNSEEN')**: Tìm kiếm các email chưa đọc trong hộp thư.
6. **if not email\_ids[0]: ...**: Kiểm tra xem có email chưa đọc nào không. Nếu không tìm thấy, in ra thông báo "No unread emails found." và kết thúc hàm.
7. **latest\_email\_id = email\_ids[0].split()[-1]**: Chọn ID của email cuối cùng trong danh sách các email chưa đọc.
8. **status, email\_data = imap.fetch(latest\_email\_id, "(RFC822)")**: Lấy dữ liệu của email cuối cùng theo ID.
9. **sender\_email, body = extract\_email\_information(email\_data[0][1])**: Trích xuất thông tin về người gửi và nội dung email từ dữ liệu email đã lấy.
10. **process\_email(sender\_email, body)**: Gọi hàm **process\_email()** để xử lý thông tin từ email như các yêu cầu được đưa ra trong nội dung email.
11. **imap.logout()**: Đóng kết nối với máy chủ IMAP, đồng thời thoát khỏi tài khoản email.

-trong file sendMail  
hàm send\_startApp:



1. **app\_controller = AppController()**: Tạo một thể hiện của lớp **AppController**. Đây có thể là một đối tượng được sử dụng để điều khiển ứng dụng trong hệ thống.
2. **subject = "APPLICATION STATUS REPORT"**: Xác định chủ đề của email sẽ được gửi.
3. **body = app\_controller.startApp(appName)**: Gọi phương thức **startApp(appName)** của **AppController** để khởi động ứng dụng có tên **appName** và thu thập thông tin về trạng thái hoạt động của ứng dụng.
4. **send\_email(subject, body, recipient\_email)**: Gọi hàm **send\_email()** để gửi email với chủ đề và nội dung đã xác định cho **recipient\_email**.

Hàm send\_endApp:



1. **app\_controller = AppController()**: Tạo một thể hiện của lớp **AppController**. Điều này giả định rằng **AppController** là một lớp hoặc đối tượng chịu trách nhiệm điều khiển hoặc quản lý các hoạt động của ứng dụng.
2. **subject = "APPLICATION STATUS REPORT"**: Đặt chủ đề email sẽ được gửi.
3. **body = app\_controller.endApp(app\_name)**: Gọi phương thức **endApp(app\_name)** của **AppController** để kết thúc ứng dụng có tên **app\_name** và thu thập thông tin về trạng thái kết thúc của ứng dụng.
4. **send\_email(subject, body, recipient\_email)**: Gọi hàm **send\_email()** để gửi email có chủ đề và nội dung đã xác định cho **recipient\_email**.