**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

A close up of a logo

Description automatically generated  
**BÁO CÁO MÔN LẬP TRÌNH MẠNG**

**GIẢNG VIÊN: BÙI DƯƠNG THẾ**

I. Tên đề tài

"Lập trình ứng dụng đọc email, gửi email"

II. Giới thiệu

Email là công cụ giao tiếp quan trọng trong công việc và cuộc sống hàng ngày. Để tối ưu hóa quy trình này, đề tài hướng đến xây dựng ứng dụng hỗ trợ tự động hóa gửi và đọc email qua các giao thức SMTP và IMAP.

Ứng dụng này gồm hai phần chính: Ứng dụng gửi email giúp người dùng lên lịch gửi email tự động kèm tệp đính kèm, và ựng dụng đọc email cho phép truy xuất và xem email từ hộp thư trong một khoảng thời gian nhất định. Với tính năng bảo mật qua mật khẩu ứng dụng, giải pháp này mang lại sự tiện lợi và hiệu quả cao trong quản lý email.

III. Giới hạn

Do đặc thù của dự án, đề tài được chia thành hai ứng dụng riêng biệt: một ứng dụng hỗ trợ gửi email và một ứng dụng hỗ trợ đọc email. Việc phân chia này giúp tối ưu hóa từng chức năng cụ thể, đồng thời dễ dàng hơn trong quá trình kiểm thử và bảo trì. Tuy nhiên, ứng dụng cũng có một số giới hạn sau:

Giới hạn dịch vụ: Chỉ hỗ trợ kết nối với dịch vụ Gmail thông qua giao thức SMTP để gửi email và IMAP để đọc email.

Phạm vi chức năng riêng lẻ: Ứng dụng gửi email không có khả năng đọc email và ngược lại, nhằm đảm bảo tính tối ưu và đơn giản trong sử dụng.

Bảo mật: Chỉ hỗ trợ đăng nhập bằng mật khẩu ứng dụng, chưa tích hợp xác thực hai yếu tố (2FA) hoặc các phương thức bảo mật nâng cao.

Quản lý hộp thư: Ứng dụng đọc email chưa hỗ trợ các tính năng nâng cao như lọc thư, phân loại thư rác hay quản lý thư mục.

IV. Mục tiêu

Xây dựng công cụ tự động hỗ trợ gửi và đọc email dễ dàng, giảm thiểu các thao tác thủ công.

Tăng cường tính linh hoạt trong việc gửi email, cho phép người dùng lên lịch gửi và kèm theo tệp đính kèm.

Đảm bảo bảo mật thông tin người dùng thông qua cơ chế sử dụng mật khẩu ứng dụng.

V. Mục đích

Giúp người dùng quản lý email hiệu quả hơn.

Tối ưu hóa công việc liên quan đến email thông qua việc giảm thiểu thời gian và thao tác thủ công.

Phát triển một ứng dụng thân thiện, đơn giản nhưng đáp ứng được các yêu cầu căn bản trong quản lý và gửi email.

VI. Lý thuyết

Gửi và đọc email là hai thao tác phổ biến và quan trọng trong truyền thông điện tử hiện nay. Việc tự động hóa các thao tác này giúp cải thiện hiệu suất công việc, đặc biệt trong các tình huống đòi hỏi gửi thông báo định kỳ hoặc xử lý email hàng loạt.

Đề tài sử dụng hai giao thức chính: SMTP và IMAP. SMTP hỗ trợ gửi email từ client đến server và từ server này đến server khác, giúp đảm bảo thông tin đến được đúng người nhận. Trong khi đó, IMAP là giao thức cho phép đọc và quản lý email từ máy chủ, giúp người dùng truy cập hộp thư của mình từ nhiều thiết bị mà không cần tải toàn bộ email về máy.

Ứng dụng này được tách thành hai phần: Gửi email và Đọc email. Việc chia tách hai chức năng này giúp tối ưu hóa xử lý và dễ dàng nâng cấp tính năng của từng phần, đồng thời giảm thiểu độ phức tạp khi người dùng chỉ cần thực hiện một trong hai thao tác.

VII. Thực hành/Thực nghiệm

* **Ứng dụng gửi Email**

1. Khai báo thư viện và khởi tạo các biến

Khai báo các thư viện cần thiết:

System.Net.Mail: Cung cấp các lớp hỗ trợ việc gửi email qua các giao thức như SMTP.

System.Threading.Tasks: Giúp thực hiện các tác vụ không đồng bộ, cần thiết để gửi email mà không làm đóng băng giao diện.

System.IO: Được dùng để kiểm tra và quản lý file đính kèm.

Các biến chính:

lstPathFile là danh sách chứa đường dẫn các file người dùng chọn để đính kèm vào email.

emailTimer là đối tượng Timer để hẹn giờ gửi email.

sendTime lưu thời gian mà người dùng thiết lập để gửi email.

2. Thiết lập Timer và DateTimePicker để chọn thời gian gửi

Trong hàm khởi tạo, Timer được thiết lập với Interval là 1000ms, nghĩa là Timer sẽ kiểm tra mỗi giây một lần. Sự kiện Tick của emailTimer sẽ gọi hàm EmailTimer\_Tick, giúp ứng dụng kiểm tra thời điểm hiện tại và tự động gửi email khi đến thời điểm đã định.

DateTimePicker được cấu hình để người dùng chọn ngày gửi theo định dạng dd/MM/yyyy. Thành phần này giúp người dùng dễ dàng chọn thời gian chính xác cho việc gửi email.

3. Xử lý sự kiện khi nhấn nút gửi email

Trong hàm xử lý sự kiện button1\_Click khi người dùng nhấn nút gửi:

Nếu chọn hẹn giờ (Set Time), thời gian được lấy từ DateTimePicker cùng các ô nhập Giờ, Phút, Giây.

Ứng dụng kiểm tra xem thời gian chọn có trong tương lai không. Nếu thời gian hợp lệ, Timer sẽ được bật (emailTimer.Start) để bắt đầu đếm ngược đến thời điểm gửi.

Nếu không chọn hẹn giờ, hàm SendEmailAsync được gọi trực tiếp để gửi email ngay lập tức.

4. Hàm gửi email SendEmailAsync

Hàm SendEmailAsync được gọi khi đến thời điểm gửi email hoặc khi người dùng gửi email ngay lập tức. Hàm này hoạt động không đồng bộ, giúp tránh việc giao diện bị treo trong lúc gửi email.

Các bước trong hàm này:

Thu thập thông tin email: Lấy địa chỉ email người gửi, người nhận, mật khẩu ứng dụng của người gửi, tiêu đề, và nội dung từ các ô nhập.

Cấu hình SMTP: Tạo một SmtpClient để gửi email qua máy chủ Gmail, với SSL và xác thực bảo mật bằng mật khẩu ứng dụng.

Tạo nội dung email: Tạo một MailMessage chứa địa chỉ người gửi, người nhận, chủ đề, và nội dung email.

Đính kèm file: Nếu có file, hàm AddAttachments sẽ thêm các file này vào email, với mỗi file đều được kiểm tra kích thước để đảm bảo tổng không vượt quá 25 MB.

Gửi email: Sử dụng smtp.SendMailAsync(mail) để gửi email và hiển thị thông báo thành công hoặc lỗi nếu có.

5. Hàm AddAttachments để thêm file đính kèm

Hàm này duyệt qua danh sách file (lstPathFile), kiểm tra tính tồn tại của từng file trước khi thêm vào email. Nếu tổng kích thước file vượt quá 25 MB, hàm sẽ hiển thị thông báo lỗi và hủy bỏ quá trình đính kèm.

6. Giao diện ứng dụng



* **Ứng dụng đọc Email**

1. Khai báo và khởi tạo kết nối với IMAP

MailKit.Net.Imap để tạo kết nối đến máy chủ Gmail và MailKit.Security để thiết lập kết nối bảo mật.

ImapClient: Dùng để kết nối và truy cập hộp thư đến.

SecureSocketOptions.SslOnConnect: Kết nối an toàn đến máy chủ Gmail với cổng 993.

2. Xử lý sự kiện nhấn nút đọc email

Trong sự kiện button1\_Click, ứng dụng thực hiện các bước:

Kết nối đến Gmail: Dùng ConnectAsync để kết nối với Gmail qua cổng 993 với SSL.

Xác thực tài khoản: AuthenticateAsync đăng nhập vào tài khoản người dùng bằng email và mật khẩu ứng dụng.

3. Mở hộp thư đến và tìm kiếm email theo thời gian

Sau khi kết nối thành công:

Mở hộp thư đến (inbox) với quyền ReadOnly để tránh xóa hay chỉnh sửa email.

Dùng SearchQuery để lọc email theo khoảng thời gian người dùng chọn:

SearchQuery.DeliveredAfter(startDate) và SearchQuery.DeliveredBefore(endDate.AddDays(1)) đảm bảo chỉ lấy email trong khoảng thời gian đã chọn.

4. Đọc và hiển thị email trên DataGridView

Với mỗi email tìm thấy, ứng dụng sẽ:

Tạo đối tượng EmailInfo lưu trữ thông tin email bao gồm người gửi, tiêu đề, nội dung, và thời gian nhận.

Thêm vào danh sách emailList và gán cho DataGridView làm nguồn dữ liệu (DataSource), hiển thị từng email theo dòng.

5. Lớp EmailInfo

EmailInfo là lớp mô tả cấu trúc của từng email, giúp dễ dàng hiển thị email trong DataGridView. Lớp này bao gồm các thuộc tính:

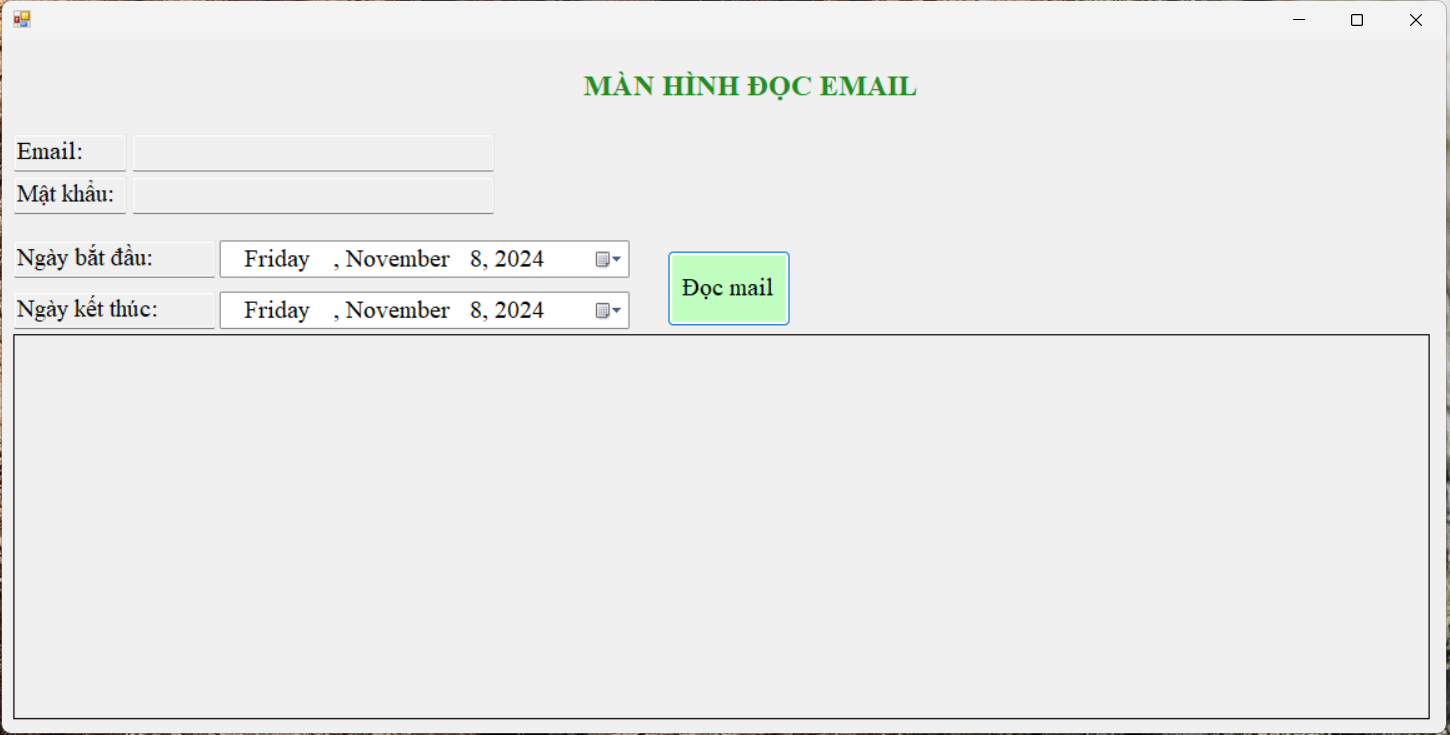
From: Địa chỉ email của người gửi.

Subject: Tiêu đề email.

body: Nội dung chính của email.

TimeReceive: Thời gian nhận email.

6. Giao diện ứng dụng



* **Ứng dụng đọc Email**

VIII. Kinh nghiệm

Khám phá các thư viện như MailKit để xử lý các giao thức gửi/nhận email.

Kinh nghiệm trong việc xử lý các lỗi kết nối, đặc biệt với các dịch vụ có yêu cầu bảo mật cao như Gmail.

Kinh nghiệm trong việc tối ưu giao diện và làm rõ thông báo lỗi để cải thiện trải nghiệm người dùng.

Kinh nghiệm trong sự hợp tác làm việc nhóm và kết nối giữa các hạng mục công việc.

IX. Kết quả đạt được

Ứng dụng hoạt động ổn định cho cả hai chức năng gửi và đọc email.

Giao diện dễ sử dụng, người dùng có thể nhập thông tin, hẹn giờ gửi email và tìm kiếm email trong khoảng thời gian nhất định.

Ứng dụng đã tự động hóa một phần đáng kể công việc quản lý email.

X. Kết quả chưa đạt được

Chưa hỗ trợ được các nhà cung cấp dịch vụ email khác ngoài Gmail.

Chưa có khả năng quản lý email nâng cao như phân loại thư rác, lọc các email theo độ ưu tiên, v.v.

Chưa tích hợp được tính năng đọc và phân loại email theo danh mục hoặc thư mục con.

Chỉ hợp tác qua hình thức online, chưa hợp tác trực tiếp offline.

Chưa có sự tổ chức kỷ luật thời gian hoàn thành dự án được như mong đợi.

XI. Hướng phát triển

Tích hợp hỗ trợ cho nhiều nhà cung cấp dịch vụ email hơn, bao gồm Yahoo, Outlook.

Phát triển thêm các chức năng lọc và phân loại email nâng cao.

Xây dựng tính năng quản lý các hộp thư và hiển thị trực quan hơn cho người dùng.

XII. Tài liệu tham khảo

[System.Net.Mail Namespace - Microsoft Learn](https://learn.microsoft.com/en-us/dotnet/api/system.net.mail)

[WebRTC Official Website](https://webrtc.org/?hl=vi)

[Microsoft .NET Documentation](https://docs.microsoft.com/en-us/dotnet/)

[Stack Overflow - .NET and Email Handling](https://stackoverflow.com/)