

BÁO CÁO THỰC HÀNH

Môn học: Lập trình mạng căn bản

Buổi báo cáo: Lab 05

Tên chủ đề: Sending and Receiving Emails in C#

GVHD: Đỗ Thị Hương Lan

Ngày thực hiện: 23/11/2024

Ngày nộp báo cáo: 29/11/2024

THÔNG TIN CHUNG:

Lớp: NT106.P11.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Lê Nhật Đăng	23520231	23520231@gm.uit.edu.vn
2	Huỳnh Minh Đạt	23520249	23520249@gm.uit.edu.vn

ĐÁNH GIÁ KHÁC:

Nội dung	Kết quả
Tổng thời gian thực hiện bài thực hành trung bình	1 tuần
Link Video thực hiện (nếu có)	
Ý kiến (nếu có) + Khó khăn + Đề xuất ...	

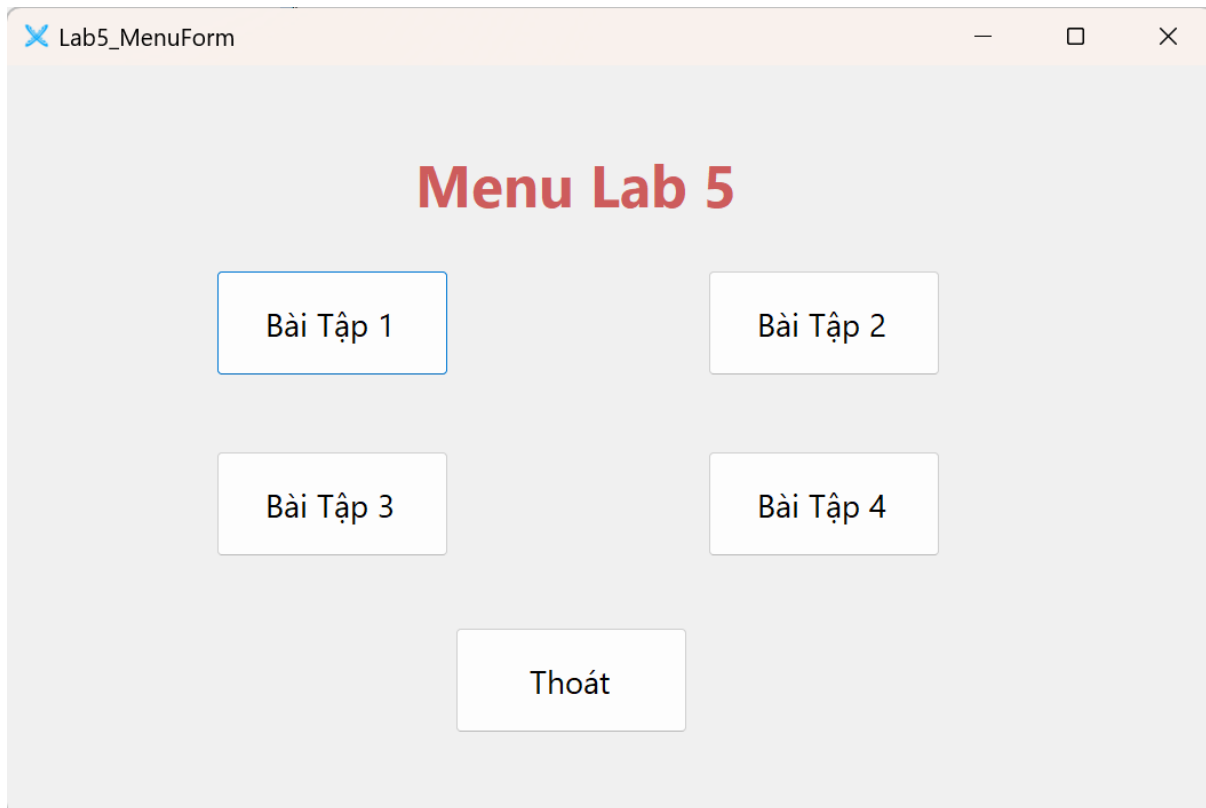
Phần bên dưới của báo cáo này là báo cáo chi tiết của nhóm thực hiện

MỤC LỤC

1. Form “Menu” của Lab05:	3
2. Bài tập 1 – Viết ứng dụng cho phép gửi mail (mail nội bộ):	4
a) Tổng quan:	4
b) Chi tiết:	4
3. Bài tập 2 – Viết ứng dụng cho phép đọc mail nội bộ (IMAP).....	8
a) Tổng quan:	8
b) Chi tiết:	8
4. Bài tập 3 – Viết ứng dụng cho phép gửi mail (via Email):	12
a) Tổng quan:	12
b) Chi tiết:	13
5. Bài tập 4 – Tổng Hợp:	17
a) Tổng quan:	17
b) Chi tiết:	18

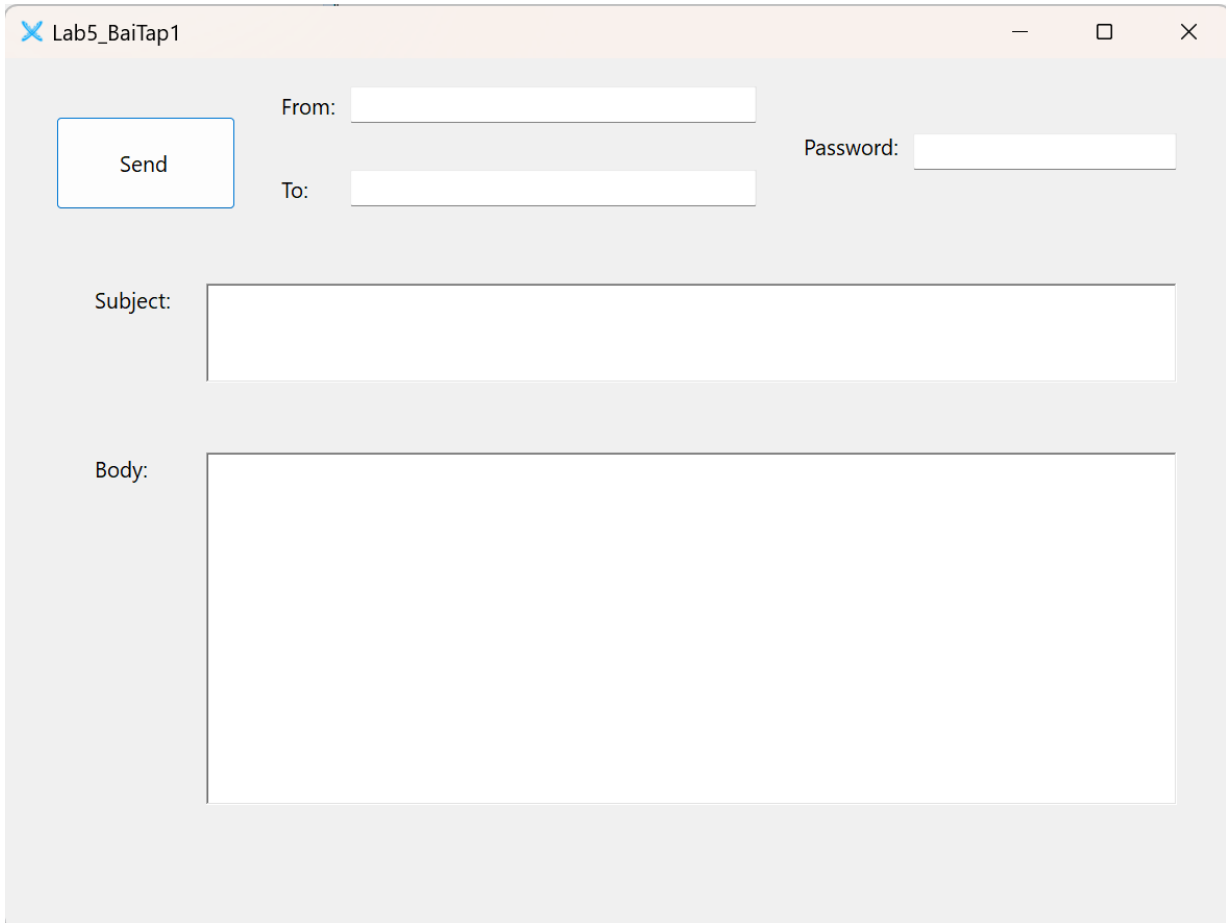
BÁO CÁO CHI TIẾT

1. Form “Menu” của Lab05:



2. Bài tập 1 – Viết ứng dụng cho phép gửi mail (mail nội bộ):

a) Tổng quan:



The screenshot shows a Windows application window titled "Lab5_BaiTap1". The window has a light gray background. On the left side, there is a blue button labeled "Send". To the right of the button, there are three input fields: "From:" (with a white text box), "To:" (with a white text box), and "Password:" (with a white text box). Below these fields, there is a "Subject:" label followed by a white text box. At the bottom, there is a "Body:" label followed by a large white text area for composing the email body.

b) Chi tiết:

- Người dùng tiến hành nhập Email người nhận (To), Email người gửi (From) và mật khẩu Email của người nhận
- Nhập chủ đề và nội dung người dùng muốn gửi mail đến người nhận.

Lab5_BaiTap1

From:

To:

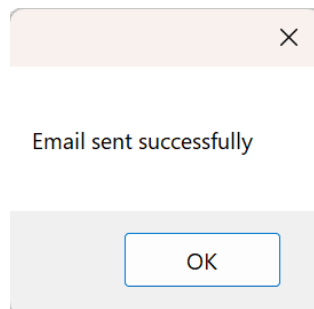
Password:

Send

Subject:

Body:

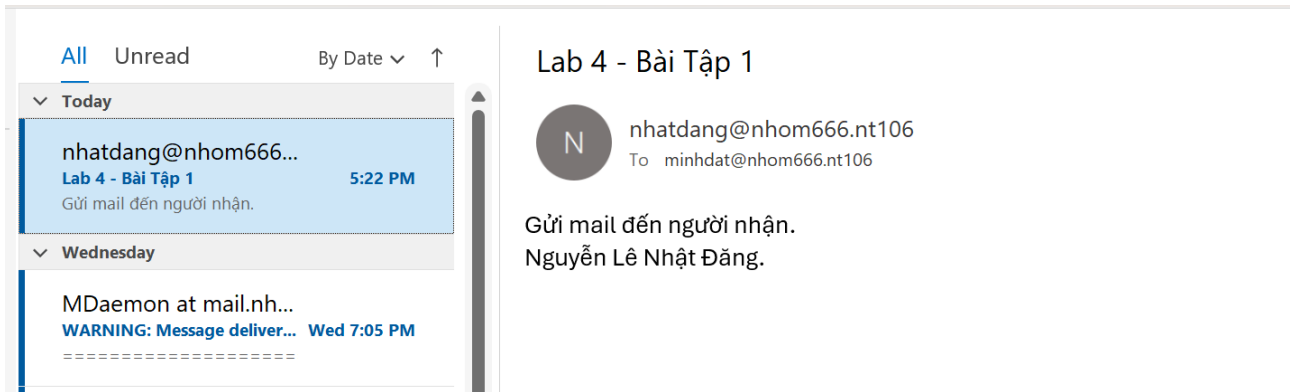
- Nhấn Send, nếu nhập đúng hết thông tin cần xác thực, chương trình sẽ trả về thông báo Email đã được gửi đi thành công



- Nếu không, chương trình sẽ báo lỗi đến người dùng (do người dùng nhập sai thông tin như mật khẩu, email gửi hoặc nhận)



- Kết quả nhận được khi gửi Email thành công (Từ Outlook Classic)



- Quá trình hoạt động từ đoạn code

```
public partial class Lab5_BaiTap1 : Form
{
    1 reference
    public Lab5_BaiTap1()
    {
        InitializeComponent();
    }

    1 reference
    private void btn_Send_Click(object sender, EventArgs e)
    {
        using (SmtpClient smtpClient = new SmtpClient("127.0.0.1"))
        {
            string mailfrom = tbx_EmailFrom.Text.ToString().Trim();
            string mailto = tbx_EmailTo.Text.ToString().Trim();
            string password = tbx_InputPassword.Text.ToString().Trim();
            var basicCredential = new System.Net.NetworkCredential(mailfrom, password);
            using (MailMessage message = new MailMessage())
            {
                MailAddress fromAddress = new MailAddress(mailfrom);
                smtpClient.UseDefaultCredentials = false;
                smtpClient.Credentials = basicCredential;

                message.From = fromAddress;
                message.Subject = rtbx_Subject.Text.ToString().Trim();
                string body = rtbx_Body.Text.ToString().Trim();
                string htmlBody = body.Replace("\n", "<br>"); // Chuyển đổi ký tự '\n' thành <br>
                message.Body = htmlBody;
                message.IsBodyHtml = true;

                message.To.Add(tbx_EmailTo.Text);
                try
                {
                    smtpClient.Send(message);
                    MessageBox.Show("Email sent successfully");
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Đã xảy ra lỗi: " + ex.Message, "Lỗi không xác định", MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
        }
    }
}
```

- Sự kiện btn_Send_Click: sự kiện gửi đi nội dung Email hoạt động
+ Khởi tạo một đối tượng SmtpClient và các thuộc tính của đối tượng liên quan

```
var basicCredential = new System.Net.NetworkCredential(mailfrom, password);
using (MailMessage message = new MailMessage())
{
    MailAddress fromAddress = new MailAddress(mailfrom);
    smtpClient.UseDefaultCredentials = false;
    smtpClient.Credentials = basicCredential;
```

- Thuộc tính UseDefaultCredentials để lấy thông tin xác thực từ hệ thống hoặc từ người dùng nhập vào thông qua tên đăng nhập và mật khẩu

```
message.From = fromAddress;
message.Subject = rtbx_Subject.Text.ToString().Trim();
string body = rtbx_Body.Text.ToString().Trim();
string htmlBody = body.Replace("\n", "<br>"); // Chuyển đổi ký tự '\n' thành <br>
message.Body = htmlBody;
message.IsBodyHtml = true;
```

```
message.To.Add(tbx_EmailTo.Text);
```

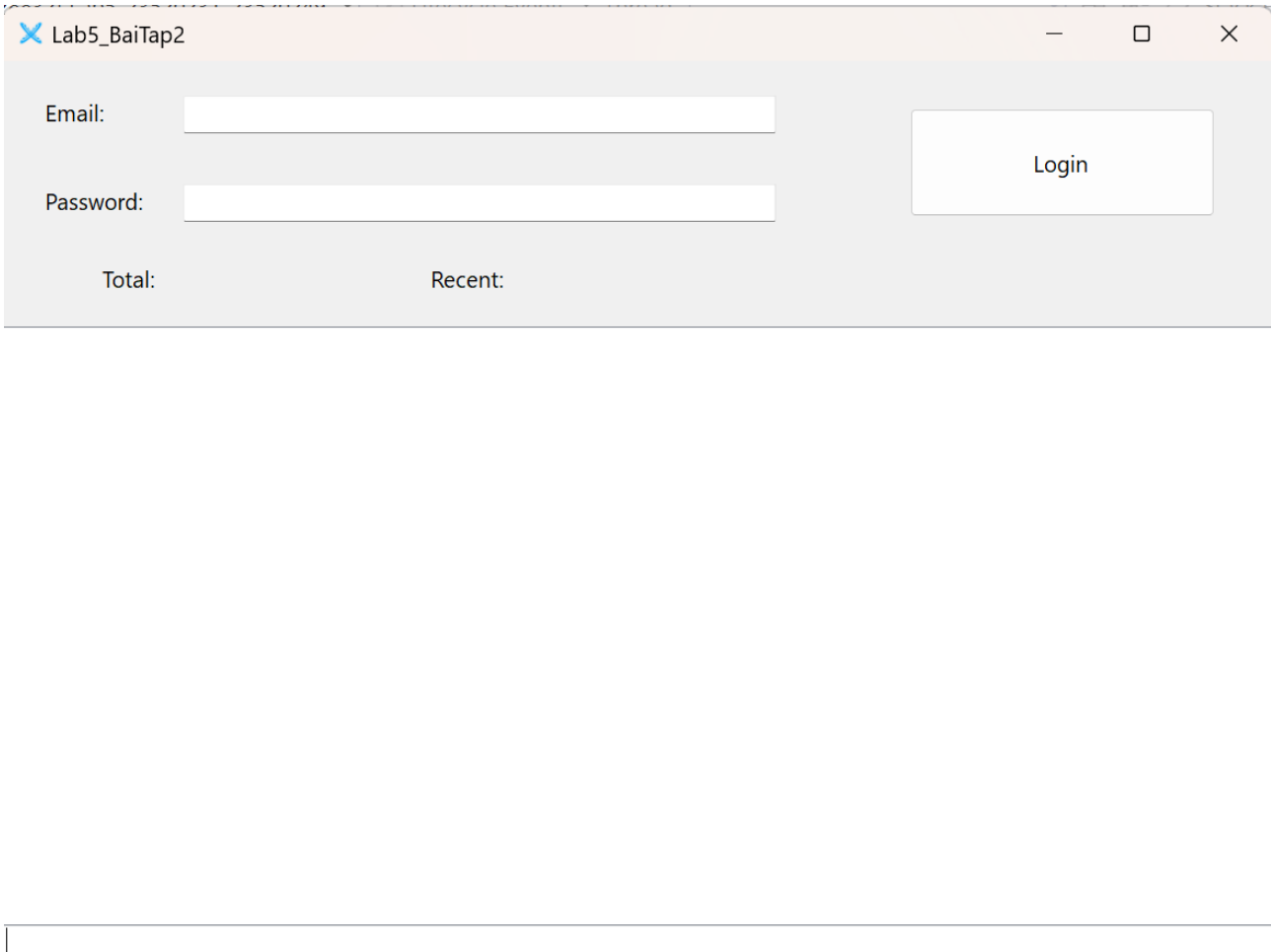
- Sau đó khởi tạo một đối tượng MailMessage để chứa thông tin email cần gửi:
 - o Thuộc tính From: để thiết lập địa chỉ email cần gửi
 - o Thuộc tính Subject & Body: để lấy thông tin về tiêu đề và nội dung của email
- Chuyển đổi ký tự '\n' đã nhận được từ textbox thể thể
 để đúng định dạng xuống dòng trong ngôn ngữ HTML.

```
try
{
    smtpClient.Send(message);
    MessageBox.Show("Email sent successfully");
}
catch (Exception ex)
{
    MessageBox.Show("Đã xảy ra lỗi: " + ex.Message, "Lỗi không xác định", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
```

- Sau khi hiện hiện phương thức Send, nếu thực hiện sẽ thông báo thành công về cho người dùng hoặc thông báo lỗi như gặp lỗi nhập liệu từ người dùng.

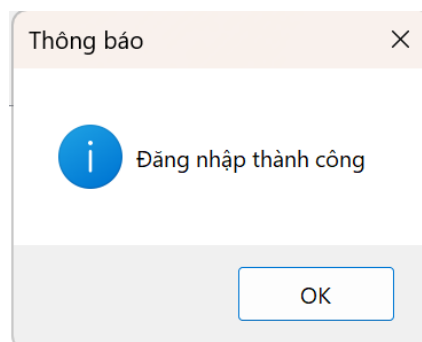
3. Bài tập 2 – Viết ứng dụng cho phép đọc mail nội bộ (IMAP).

a) Tổng quan:

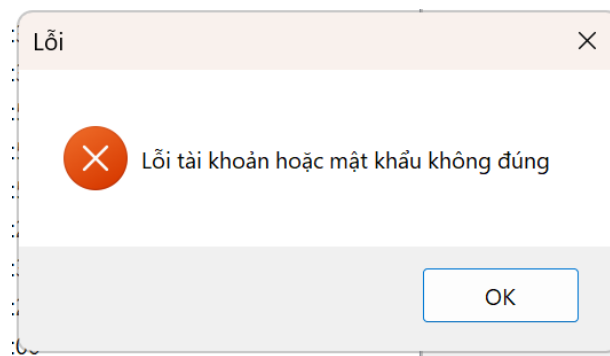


b) Chi tiết:

- Người dùng tiến hành nhập Email và Password và tiến hành Login, chương trình sẽ hiển thị thông báo đăng nhập thành công, sau đó các nội dung Mail sẽ hiện ra.



- Trường hợp nhập sai thông tin từ phía người dùng



- Tổng quan giao diện đọc mail:

Lab5_BaiTap2

Email: minhdat@nhom666.nt106

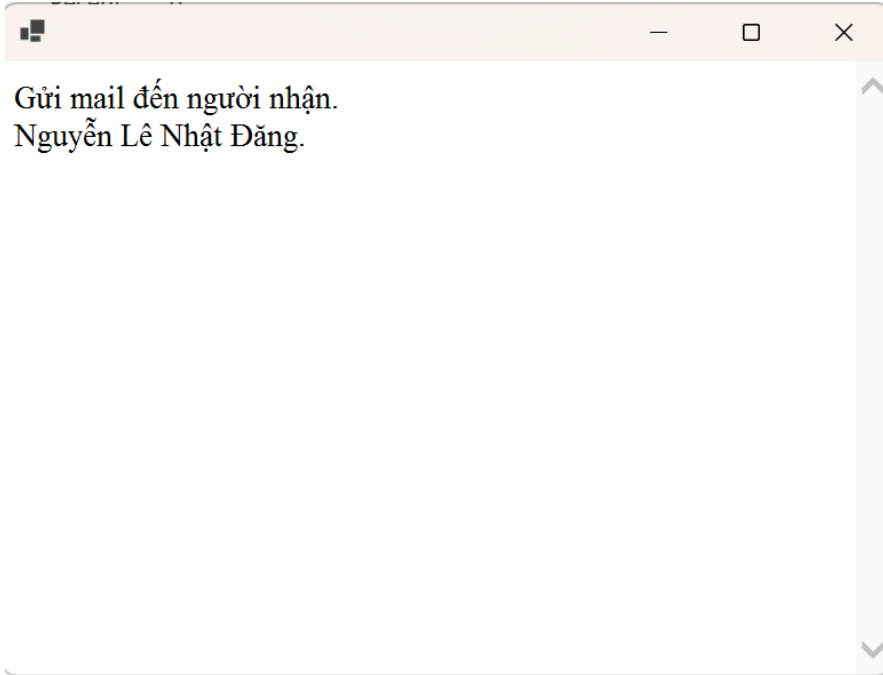
Password: ●●●●●●●●

Login

Total: 13 Recent: 0

Email	From	Thời gian
Lab 4 - Bài Tập 1	nhatdang@nhom666.nt...	30/11/2024 17:21:56
WARNING: Message delivery failed	"MDaemon at mail.nho...	27/11/2024 19:04:30
WARNING: Message delivery failed	"MDaemon at mail.nho...	27/11/2024 19:04:30
WARNING: Message delivery failed	"MDaemon at mail.nho...	27/11/2024 19:04:30
WARNING: Message delivery delayed	"MDaemon at mail.nho...	25/11/2024 15:27:59
WARNING: Message delivery delayed	"MDaemon at mail.nho...	25/11/2024 15:27:59
WARNING: Message delivery delayed	"MDaemon at mail.nho...	25/11/2024 15:27:59
Hello	nhatdang@nhom666.nt...	25/11/2024 15:13:26
Hello	nhatdang@nhom666.nt...	25/11/2024 15:08:36
Welcome to the MDAemon email system...	"MDaemon at mail.nho...	18/11/2024 09:32:21
Microsoft Outlook Test Message	"Microsoft Outlook" <m...	01/01/0001 00:00:00
test mail from Telv2	nhatdang@nhom666.nt...	01/01/0001 00:00:00
Test mail from Telnet v2	nhatdang@nhom666.nt...	01/01/0001 00:00:00

- Click double vào mail cần xem thông tin, cửa sổ mới sẽ hiện ra.



Gửi mail đến người nhận.
Nguyễn Lê Nhật Đăng.

- Giải thích:

```
// Thêm một biến toàn cục để lưu danh sách email đã sắp xếp
private List<MimeMessage> sortedEmails = new List<MimeMessage>();
1 reference
private void btn_Login_Click(object sender, EventArgs e)
{
    var client = new MailKit.Net.Imap.ImapClient();
    client.ServerCertificateValidationCallback = (s, c, h, e) => true;
    client.Connect("127.0.0.1", 993);
    try
    {
        client.Authenticate(tbx_Email.Text, tbx_Password.Text);
    }
    catch (AuthenticationException)
    {
        MessageBox.Show("Lỗi tài khoản hoặc mật khẩu không đúng", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    MessageBox.Show("Đăng nhập thành công", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);
    var inbox = client.Inbox;
    inbox.Open(MailKit.FolderAccess.ReadOnly);
    // Lấy danh sách email và lưu vào sortedEmails
    sortedEmails = inbox.Cast<MimeMessage>().OrderByDescending(message => message.Date).ToList();
    // Lấy danh sách thư từ hộp thư inbox
    listView.Clear();
    listView.Columns.Add("Email", 400);
    listView.Columns.Add("From", 250);
    listView.Columns.Add("Thời gian", 250);
    listView.View = View.Details;
    // Cập nhật số mail Total và Recent
    lbl_NumTotal.Text = inbox.Count.ToString();
    lbl_NumRecent.Text = inbox.Recent.ToString();
}
```

```
// Lấy danh sách thư email và sắp xếp theo thời gian gửi
var sortedMessages = inbox.OrderByDescending(message => message.Date);
foreach (var message in sortedMessages)
{
    ListViewItem name = new ListViewItem(message.Subject);

    ListViewItem.ListViewSubItem from = new ListViewItem.ListViewSubItem(name, message.From.ToString());
    name.SubItems.Add(from);

    ListViewItem.ListViewSubItem date = new ListViewItem.ListViewSubItem(name, message.Date.ToString("dd/MM/yyyy HH:mm:ss"));
    name.SubItems.Add(date);

    listView.Items.Add(name);
}
client.Disconnect(true);

private void ReadMail(object sender, EventArgs e)
{
    var client = new MailKit.Net.Imap.ImapClient();
    client.ServerCertificateValidationCallback = (s, c, h, e_ssl) => true;
    client.Connect("127.0.0.1", 993);
    try
    {
        client.Authenticate(tbx_Email.Text, tbx_Password.Text);
    }
    catch (AuthenticationException)
    {
        MessageBox.Show("Lỗi tài khoản hoặc mật khẩu không đúng", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    var index = listView.FocusedItem.Index;

    // Truy cập email trong danh sách đã sắp xếp
    var message = sortedEmails[index];
    // Lấy danh sách thư từ hộp thư Inbox
    var inbox = client.Inbox;
    inbox.Open(MailKit.FolderAccess.ReadOnly);
    // Hiển thị nội dung HTML của email trong WebBrowser Control
    var webBrowserForm = new Form();
    var webBrowser = new WebBrowser();
    webBrowserForm.Controls.Add(webBrowser);
    webBrowser.Dock = DockStyle.Fill;
    webBrowser.DocumentText = string.IsNullOrEmpty(message.TextBody) ? message.HtmlBody : message.TextBody;
    webBrowserForm.Size = new Size(800, 600);

    webBrowserForm.ShowDialog();

    client.Disconnect(true);
}
```

- Kết nối IMAP:

```
var client = new MailKit.Net.Imap.ImapClient();
client.ServerCertificateValidationCallback = (s, c, h, e) => true;
client.Connect("127.0.0.1", 993);
```

- Tạo kết nối ImapClient để kết nối đến server IMAP.
- Kết nối đến server IMAP ở địa chỉ 127.0.0.1 ở cổng 993
- Xác thực người dùng từ textbox Email và Password và xử lý trường hợp lỗi nếu có.

```
try
{
    client.Authenticate(tbx_Email.Text, tbx_Password.Text);
}
catch (AuthenticationException)
{
    MessageBox.Show("Lỗi tài khoản hoặc mật khẩu không đúng", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    return;
}
MessageBox.Show("Đăng nhập thành công", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

- Lấy danh sách email: Truy cập hộp thư inbox và sắp xếp email theo ngày gửi giảm dần, lưu vào sortedEmails (list được định nghĩa để lưu trữ các Mail đã sắp xếp theo mới nhất)

```
// Lấy danh sách email và lưu vào sortedEmails
sortedEmails = inbox.Cast<MimeMessage>().OrderByDescending(message => message.Date).ToList();
```

- Hiển thị email trong listView (code giảng viên đã hướng dẫn)
- Sự kiện read mail, tiến hành đọc mail khi người dùng nhấn vào mail trong danh sách để xem nội dung chi tiết

- Xác thực lại giống theo tác lúc đăng nhập
- Lấy email được chọn
- Hiển thị nội dung Email và ngắt kết nối IMAP

```
// Truy cập email trong danh sách đã sắp xếp
var message = sortedEmails[index];
// Lấy danh sách thư từ hộp thư Inbox
var inbox = client.Inbox;
inbox.Open(MailKit.FolderAccess.ReadOnly);
// Hiển thị nội dung HTML của email trong WebBrowser Control
var webBrowserForm = new Form();
var webBrowser = new WebBrowser();
webBrowserForm.Controls.Add(webBrowser);
webBrowser.Dock = DockStyle.Fill;
webBrowser.DocumentText = string.IsNullOrEmpty(message.TextBody) ? message.HtmlBody : message.TextBody;
webBrowserForm.Size = new Size(800, 600);
```

```
webBrowserForm.ShowDialog();
```

```
client.Disconnect(true);
```

4. Bài tập 3 – Viết ứng dụng cho phép gửi mail (via Email):

a) Tổng quan:

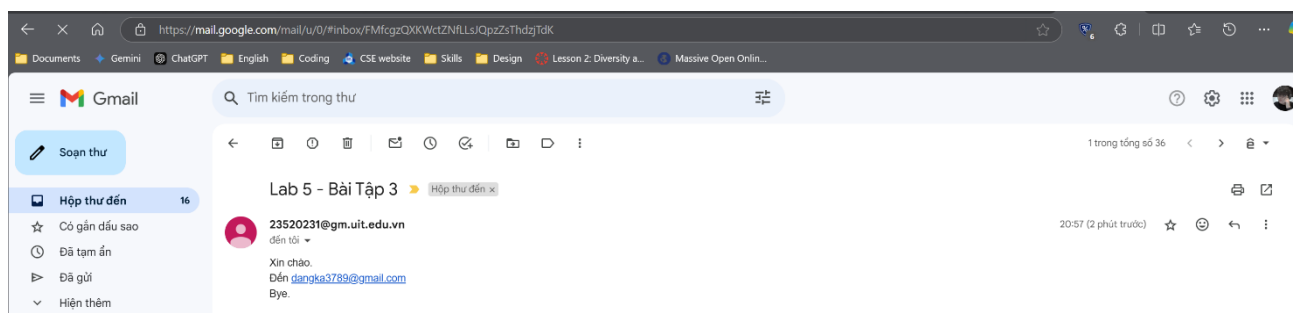
The screenshot shows a Windows application window titled "Form1". Inside the window, there are several input fields and a button:

- From:** A text input field.
- Password:** A text input field.
- To:** A text input field.
- Subject:** A text input field.
- Body:** A large multi-line text area for composing the email body.
- Send:** A button located at the bottom right of the form, used to send the email.

b) Chi tiết:

- Tương tự bài 1, người dùng tiến hành nhập email người gửi và email người nhận, Subject & Body cần gửi.
- Lưu ý: Password được dùng cho người gửi ở dạng mật khẩu được lấy từ App Password do người dùng tạo ở phần Bảo Mật – Google.
- Thông báo Email đã được gửi thành công.

- Kết quả nhận được:



- Chi tiết đoạn code hoạt động

```

public partial class Lab5_BaiTap3 : Form
{
    1 reference
    public Lab5_BaiTap3()
    {
        InitializeComponent();
    }

    1 reference
    private void btn_Send_Click(object sender, EventArgs e)
    {
        string senderEmail = tbx_EmailFrom.Text;
        string senderPassword = tbx_InputPassword.Text;
        string recipientEmail = tbx_EmailTo.Text;
        string subject = rtbx_Subject.Text;
        string body = rtbx_Body.Text.Replace("\n", Environment.NewLine);

        if (!CheckValidEmail(senderEmail)) //Kiểm tra tính hợp lệ của email qua hàm CheckValidEmail
        {
            MessageBox.Show("Địa chỉ email người gửi không hợp lệ.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
        }
        if (!CheckValidEmail(recipientEmail))
        {
            MessageBox.Show("Địa chỉ email người nhận không hợp lệ.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
        }
        if (string.IsNullOrEmpty(senderPassword))//Kiểm tra password
        {
            MessageBox.Show("Mật khẩu không được để trống.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
        }
        if (string.IsNullOrEmpty(body))//Kiểm tra nếu nội dung rỗng
        {
            MessageBox.Show("Nội dung email rỗng.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning); return;
        }

        try
        {
            //Tạo SmtpClient để gửi mail thông qua SMTP
            SmtpClient client = new SmtpClient("smtp.gmail.com", 587)
            {
                Credentials = new NetworkCredential(senderEmail, senderPassword), //Thông tin bảo mật
                EnableSsl = true //Kích hoạt SSL để bảo mật kết nối
            };
            MailMessage mailMessage = new MailMessage
            {
                From = new MailAddress(senderEmail),
                Subject = subject, //Chủ đề
                Body = body //Nội dung
            };
            mailMessage.To.Add(recipientEmail);
            client.Send(mailMessage); //Gửi mail
            MessageBox.Show("Email đã được gửi thành công!", "Thành công", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        catch (Exception ex) //Thông báo lỗi nếu có
        {
            MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }

    private bool CheckValidEmail(string email)
    {
        try
        {
            var check = new MailAddress(email);
            return check.Address == email;
        }
        catch
        {
            return false;
        }
    }
}

```

```
//Tạo SmtpClient để gửi mail thông qua SMTP
SmtpClient client = new SmtpClient("smtp.gmail.com", 587)
{
    Credentials = new NetworkCredential(senderEmail, senderPassword), //Thông tin bảo mật
    EnableSsl = true //Kích hoạt SSL để bảo mật kết nối
};
```

- Ở sự kiện "Send" gửi mail đến người nhận, tiến hành cấu hình SMTP Client: Thực hiện kết nối đến máy chủ SMTP của gmail "smtp.gmail.com" ở cổng 587.
- Thực hiện xác thực bằng Email và mật khẩu mà người dùng đã nhập được.
- Bật mã hoá SSL để bảo mật kết nối.

```
MailMessage mailMessage = new MailMessage
{
    From = new MailAddress(senderEmail),
    Subject = subject, //Chủ đề
    Body = body //Nội dung
};
mailMessage.To.Add(recipientEmail);
client.Send(mailMessage); //Gửi mail
MessageBox.Show("Email đã được gửi thành công!", "Thành công", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

- Tạo email với MailMessage gồm các thành phần (From, Subject, Body, To.Add)
- Thông báo email đã được gửi thành công nếu email được gửi đi thành công

```
catch (Exception ex) //Thông báo lỗi nếu có
{
    MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
```

- Hiển thị lỗi cho người dùng nếu gặp lỗi.

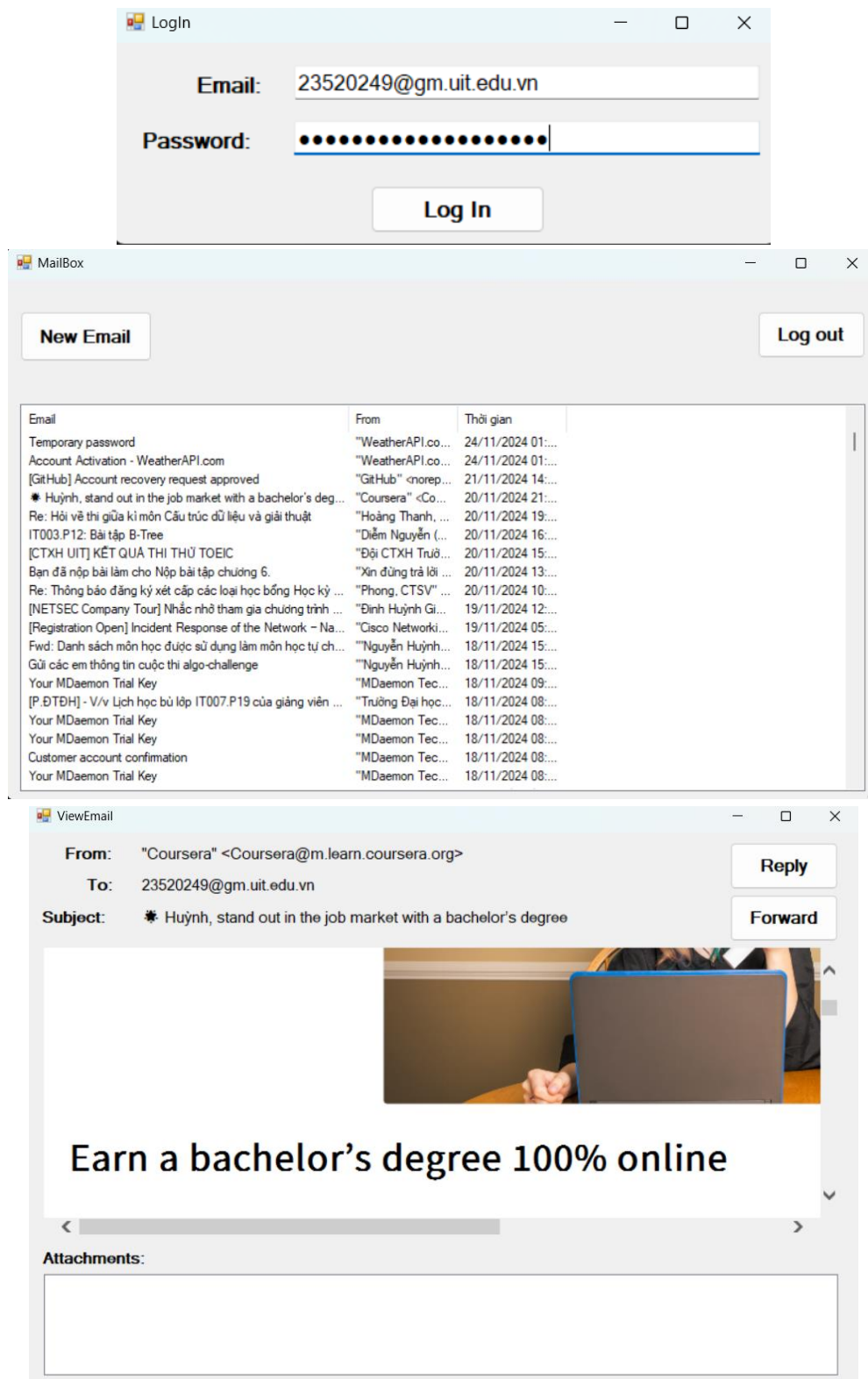
```
private bool CheckValidEmail(string email)
{
    try
    {
        var check = new MailAddress(email);
        return check.Address == email;
    }
    catch
    {
        return false;
    }
}
```

```
if (!CheckValidEmail(senderEmail)) //Kiểm tra tính hợp lệ của email qua hàm CheckValidEmail
{
    MessageBox.Show("Địa chỉ email người gửi không hợp lệ.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
}
if (!CheckValidEmail(recipientEmail))
{
    MessageBox.Show("Địa chỉ email người nhận không hợp lệ.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
}
if (string.IsNullOrEmpty(senderPassword)) //Kiểm tra password
{
    MessageBox.Show("Mật khẩu không được để trống.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
}
if (string.IsNullOrEmpty(body)) //Kiểm tra nếu nội dung rỗng
{
    MessageBox.Show("Nội dung email rỗng.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning); return;
}
```

- Trước khi gửi, kiểm tra tính hợp lệ của Gmail và thông báo cho người dùng nếu sai.

5. Bài tập 4 – Tổng Hợp:

a) Tổng quan:



b) Chi tiết:

- Bài tập 4 được chia thành 4 form riêng cho từng chức năng chính của một công cụ duyệt mail bao gồm form đăng nhập, form duyệt email, form xem email và form gửi email.
- Người dùng đăng nhập bằng địa chỉ email và app password được cài đặt từ trước qua Gmail.

```

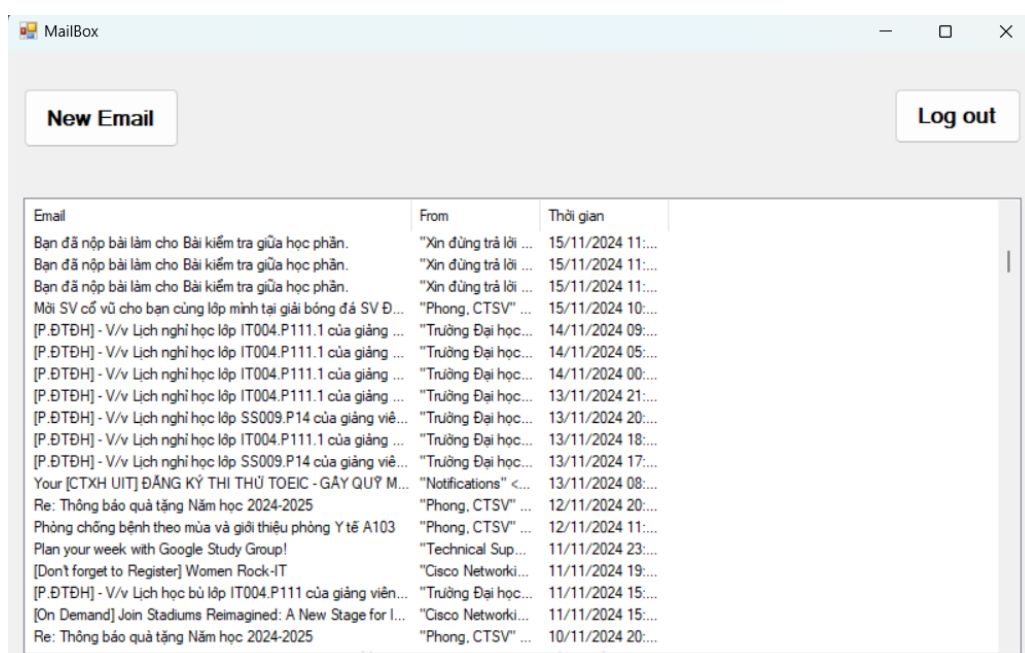
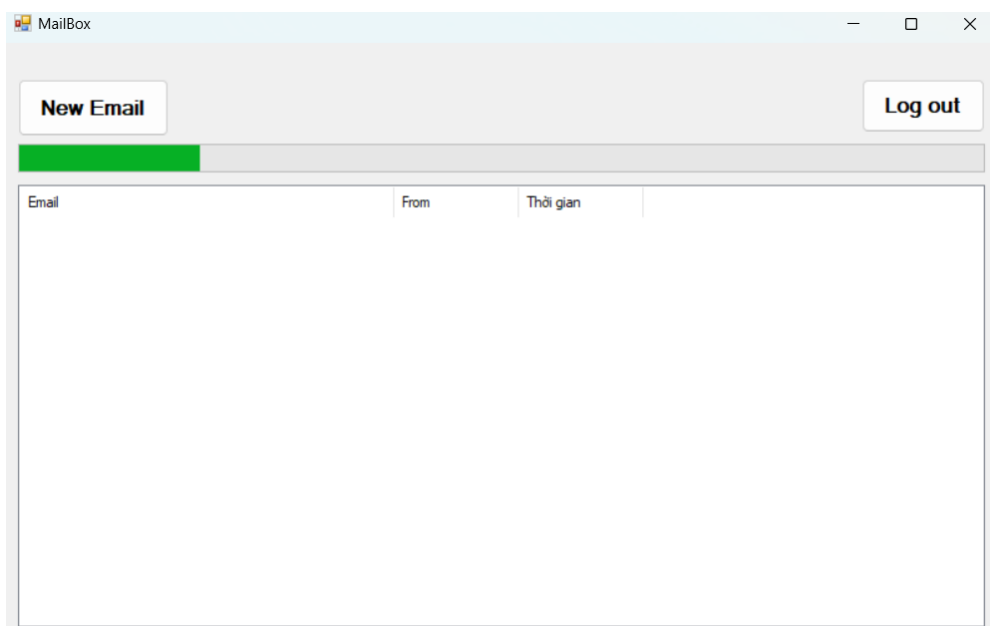
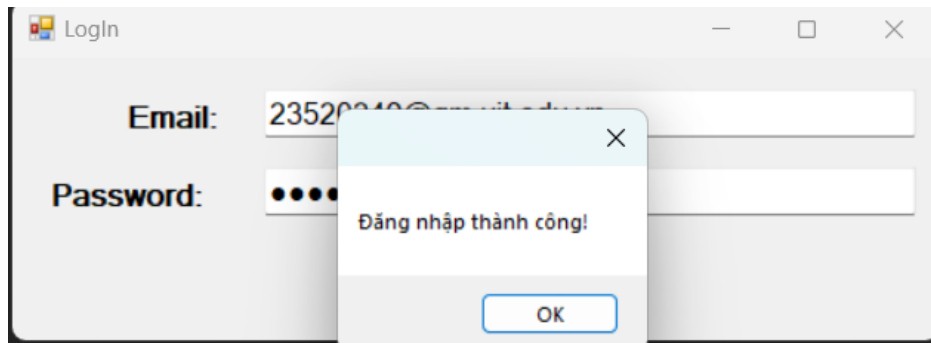
public Login()
{
    InitializeComponent();
    tb_pass.UseSystemPasswordChar = true;
}

// reference
private void bt_Login_Click(object sender, EventArgs e)
{
    try
    {
        Username = tb_email.Text;
        Password = tb_pass.Text;
        IMAPHost = "imap.gmail.com";
        IMAPPort = 993;
        SMTPHost = "smtp.gmail.com";
        SMTPPort = 465;
        client = new ImapClient();
        client.Connect(IMAPHost, IMAPPort, true);
        client.Authenticate(Username, Password);
        MessageBox.Show("Đăng nhập thành công!");
        Explorer mailBox = new Explorer();
        this.Hide();
        mailBox.ShowDialog();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

private void LogIn_FormClosed(object sender, FormClosedEventArgs e)
{
    Application.Exit();
}

```

- Nếu đăng nhập thành công và không có thông báo lỗi, giao diện hộp thư sẽ hiển thị và tải các email, nơi người dùng có thể chọn email trong listView để xem chi tiết, chọn nút New Email để soạn thảo thư mới hoặc nút Log out nếu muốn đăng nhập và làm việc ở một tài khoản khác.



```
private void DisplayEmails()
{
    listView_inbox.Items.Clear();
    foreach (var email in emails)
    {
        ListViewItem item = new ListViewItem(email.Subject);
        item.SubItems.Add(new ListViewItem.ListViewSubItem(item, email.From));
        item.SubItems.Add(new ListViewItem.ListViewSubItem(item, DateTime.Parse(email.Date).ToString("dd/MM/yyyy HH:mm:ss")));
        listView_inbox.Items.Add(item);
    }
}

//Xử lý khi người dùng chọn một email trong danh sách
1 reference
private void listView_inbox_ItemActivate(object sender, EventArgs e)
{
    if (listView_inbox.SelectedItems.Count > 0)
    {
        int selectedIndex = listView_inbox.SelectedIndex;
        if (selectedIndex >= 0 && selectedIndex < emails.Count)
        {
            Email selectedEmail = emails[selectedIndex];
            string subject = selectedEmail.Subject;
            string from = selectedEmail.From;
            string to = username;
            string date = selectedEmail.Date;
            string textBody = selectedEmail.TextBody;
            string htmlBody = selectedEmail.HtmlBody;
            List<MimeEntity> attachments = selectedEmail.Attachments;
            //Mở form hiển thị chi tiết email và truyền thông tin email vào form
            Mail_Show show = new Mail_Show(subject, from, to, date, textBody, htmlBody, attachments);
            show.Show();
        }
    }
}
```

```
private async void MailBox_Load(object sender, EventArgs e)
{
    //Hiển thị ProgressBar khi bắt đầu tải email
    progressBar.Visible = true;

    listView_inbox.Columns.Add("Email", 300);
    listView_inbox.Columns.Add("From", 100);
    listView_inbox.Columns.Add("Thời gian", 100);
    listView_inbox.View = View.Details;

    try
    {
        var inbox = client.Inbox;
        inbox.Open(FolderAccess.ReadOnly);

        emails.Clear(); //Xóa nội dung cũ

        for (int i = 0; i < inbox.Count; i++)
        {
            var message = inbox.GetMessage(i);
            var email = new Email
            {
                Subject = message.Subject,
                From = message.From.ToString(),
                Date = message.Date.ToString(),
                TextBody = message.TextBody,
                HtmlBody = message.HtmlBody
            };
            foreach (var attachment in message.Attachments)
            {
                email.Attachments.Add(attachment);
            }
            emails.Add(email);

            //Cập nhật giá trị ProgressBar dựa trên tiến độ tải email
            progressBar.Value = (int)((float)(i + 1) / inbox.Count) * 100;
            await Task.Delay(10);
        }

        // Sắp xếp danh sách email theo thời gian giảm dần
        emails = emails.OrderByDescending(email => DateTime.Parse(email.Date)).ToList();

        // Hiển thị email trong listView
        DisplayEmails();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }

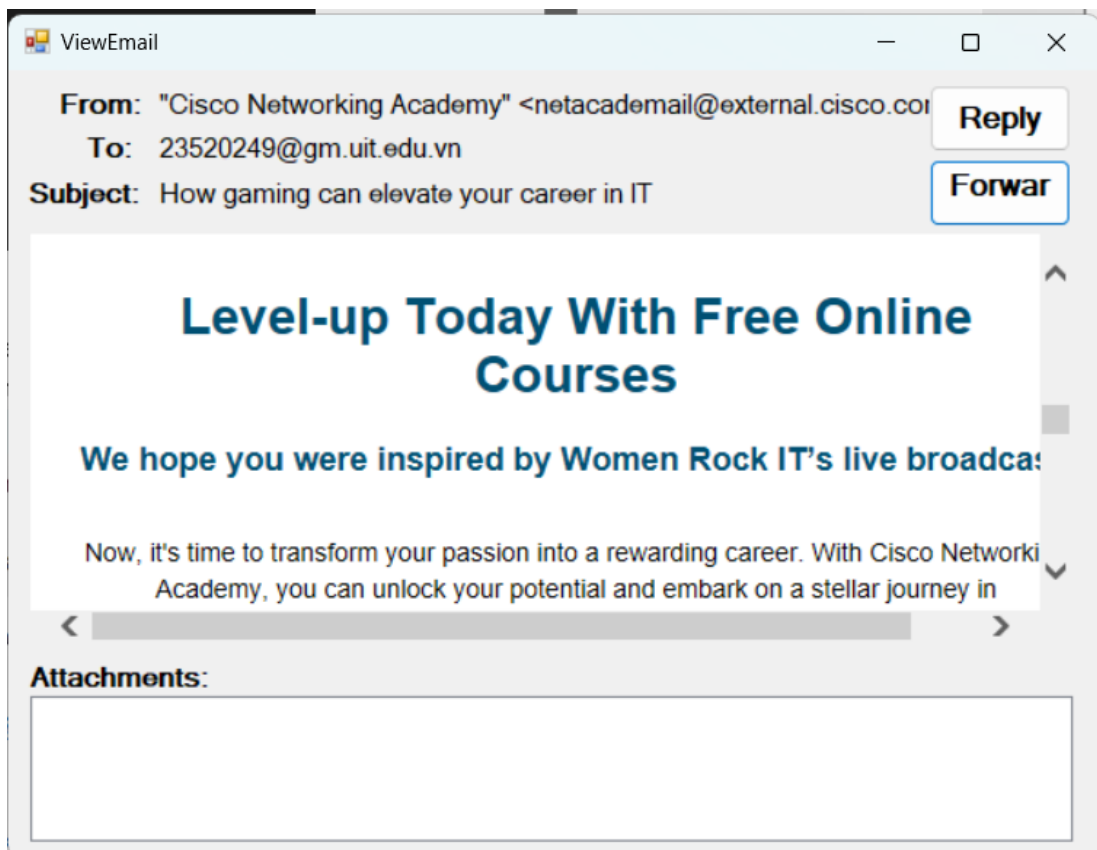
    // Ẩn ProgressBar khi quá trình tải email hoàn tất
    progressBar.Visible = false;
}
```

```
//Xử lý khi người dùng nhấn nút new mail
1 reference
private void bt_newmail_Click(object sender, EventArgs e)
{
    string username = this.username;
    string password = this.password;
    string SMTPHostName = SMTPHost;
    Int32 SMTPHostPort = SMTPPort;
    //Mở form gửi email với thông tin đăng nhập hiện tại
    Mail_Send send = new Mail_Send(username, password, SMTPHostName, SMTPHostPort);
    send.Show();
}

1 reference
private void LogOutBtn_Click(object sender, EventArgs e)
{
    client.Disconnect(true);
    this.Hide();
    Login loginForm = new Login();
    loginForm.Show();
}

1 reference
private void MailBox_FormClosed(object sender, FormClosedEventArgs e)
{
    Application.Exit();
}
```

- Người dùng chọn một Email trong list view thì email sẽ được hiển thị trong form mới với địa chỉ email người gửi, chủ đề, nội dung và tệp đính kèm nếu có.



- Người dùng có thể chọn Reply hoặc Forward để trả lời hoặc chuyển tiếp Email qua form gửi mail được tạo mới với một số nội dung sẵn.

```
//Khởi tạo của form với các thông tin chi tiết của email
1 reference
public Mail_Show(string subject, string from, string to, string date, string textBody, string htmlBody, List<MimeEntity> attachments)
{
    InitializeComponent();

    //Gán các thông tin email
    fromLabel.Text = from;
    toLabel.Text = to;
    subjectLabel.Text = subject;

    //Hiển thị nội dung email theo dạng HTML hoặc plain text
    if (!string.IsNullOrEmpty(htmlBody))
    {
        bodyText.Visible = false;
        webBrowser1.Visible = true;
        webBrowser1.DocumentText = htmlBody;
    }
    else
    {
        webBrowser1.Visible = false;
        bodyText.Visible = true;
        bodyText.Text = textBody;
    }

    //Hiển thị danh sách tệp đính kèm nếu có
    if (attachments.Count > 0)
    {
        foreach (var attachment in attachments)
        {
            ListViewItem item = new ListViewItem(attachment.ContentDisposition?.FileName ?? attachment.ContentType.Name);
            item.Tag = attachment;
            attachmentsListView.Items.Add(item);
        }
    }
}
```

```

//Xử lý khi người dùng chọn một tệp đính kèm trong danh sách
1 reference
private void attachmentsListView_ItemActivate(object sender, EventArgs e)
{
    if (attachmentsListView.SelectedItems.Count > 0)
    {
        var attachment = attachmentsListView.SelectedItems[0].Tag as MimeEntity;
        if (attachment != null)
        {
            var fileName = attachment.ContentDisposition?.FileName ?? attachment.ContentType.Name;
            var filePath = Path.Combine(Path.GetTempPath(), fileName);

            //Tạo tệp tạm thời để lưu tệp đính kèm
            using (var stream = File.Create(filePath))
            {
                if (attachment is MimePart part)
                {
                    part.Content.DecodeTo(stream);
                }
                else if (attachment is MessagePart rfc822)
                {
                    rfc822.Message.WriteTo(stream);
                }
            }

            //Mở tệp đính kèm
            System.Diagnostics.Process.Start(filePath);
        }
    }
}

```

```

//Xử lý khi người dùng nhấn nút reply
1 reference
private void bt_reply_Click(object sender, EventArgs e)
{
    //Lấy thông tin từ email gốc
    string originalSender = fromLabel.Text;
    string replyTo = toLabel.Text;
    string originalSubject = subjectLabel.Text;
    string replySubject = $"Re: {originalSubject}";

    //Lấy nội dung email gốc
    string originalBody;
    if (webBrowser1.Visible)
    {
        originalBody = webBrowser1.Text;
    }
    else
    {
        originalBody = bodyText.Text;
    }

    //Tạo nội dung trả lời
    string quotedBody = $"<div>--- Tin nhắn gốc ---</div><div>\nFrom: {originalSender}</div>\nTo: {replyTo}</div>\nSubject: {originalSubject}</div>\n\n{originalBody}</div>";

    //Mở form gửi email với các thông tin trả lời
    Mail_Send sendEmailForm = new Mail_Send(
        username: replyTo,
        password: Login.Password,
        SMTPHostName: "smtp.gmail.com",
        SMTPHostPort: 465
    );

    //Thêm các tệp đính kèm
    foreach (var attachment in attachmentsListView.Items)
    {
        var attachmentItem = attachment as ListViewItem;
        if (attachmentItem != null)
        {
            var attachmentEntity = attachmentItem.Tag as MimePart;
            if (attachmentEntity != null)
            {
                sendEmailForm.rtb_attach.Text += attachmentEntity.FileName + ";";
            }
        }
    }

    //Gán các giá trị
    sendEmailForm.tb_rcver.Text = ExtractGmailAccount(originalSender);
    sendEmailForm.tb_subject.Text = replySubject;
    sendEmailForm.rtb_body.Text = quotedBody;

    sendEmailForm.Show();
}

```

```

//Xử lý khi người dùng nhấn nút forward
1 reference
private void bt_forward_Click(object sender, EventArgs e)
{
    //Lấy thông tin từ email gốc
    string senderEmail = toLabel.Text;
    string subject = "Fwd: " + subjectLabel.Text;
    string to = "";
    string textBody = "----- Forwarded message -----\n" +
        "From: " + fromLabel.Text + "\n" +
        "To: " + toLabel.Text + "\n" +
        "Subject: " + subjectLabel.Text + "\n\n";

    //Mở form gửi email với các thông tin chuyển tiếp
    Mail_Send send = new Mail_Send(
        username: senderEmail,
        password: Login.Password,
        SMTPHostName: "smtp.gmail.com",
        SMTPHostPort: 465
    );

    //Thêm các tệp đính kèm
    foreach (var attachment in attachmentsListView.Items)
    {
        var attachmentItem = attachment as ListViewItem;
        if (attachmentItem != null)
        {
            var attachmentEntity = attachmentItem.Tag as MimePart;
            if (attachmentEntity != null)
            {
                send.rtb_attach.Text += attachmentEntity.FileName + ";";
            }
        }
    }

    //Gán các giá trị
    send.tb_subject.Text = subject;
    send.rtb_body.Text = textBody;

    send.Show();
}

```

```

1 reference
string ExtractGmailAccount(string originalSender)
{
    // Kiểm tra xem chuỗi originalSender có chứa ký tự "<" và ">" hay không
    int startIndex = originalSender.IndexOf('<');
    int endIndex = originalSender.IndexOf('>');

    if (startIndex != -1 && endIndex != -1 && endIndex > startIndex)
    {
        return originalSender.Substring(startIndex + 1, endIndex - startIndex - 1);
    }
    else
    {
        return originalSender;
    }
}

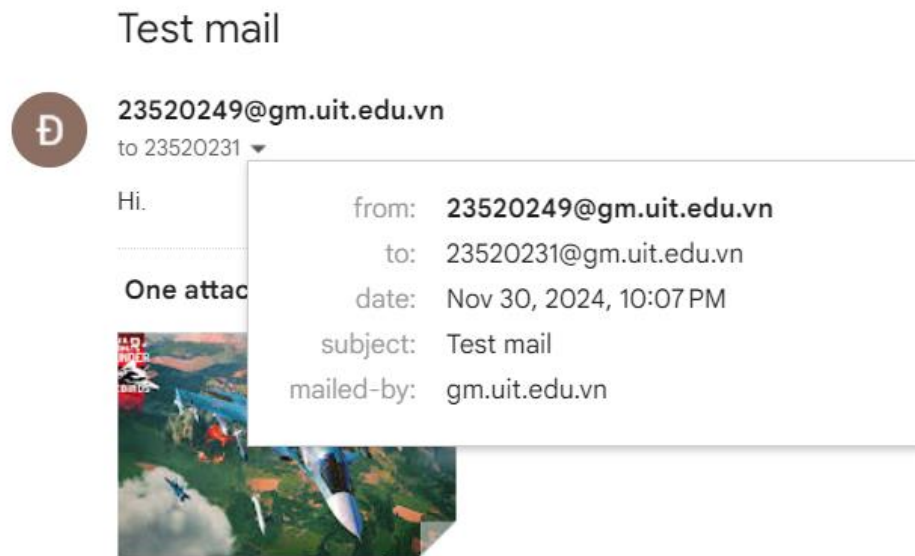
```


- Form gửi mail được cải tiến từ bài tập 3 với chức năng mới là tệp đính kèm, nếu gửi mail thành công thì người dùng sẽ nhận được thông báo.

The screenshot shows a Windows application window titled "SendEmail". It contains the following fields and controls:

- From:** 23520249@gm.uit.edu.vn
- To:** 23520231@gm.uit.edu.vn
- Subject:** Test mail
- Text Area:** Contains the text "Hi."
- HTML Checkbox:** An unchecked checkbox labeled "HTML".
- Attachments:** A section with a "Brows" button and a text box containing the file path: C:\Users\Admin\Downloads\3840x2160_firebirds_su_34_logo_eng_e8164549a84c642978cd2f74395d872b.jpg
- Send Button:** A large button at the bottom center labeled "Send".

This screenshot shows the same "SendEmail" form as above, but with a small dialog box overlaid in the center. The dialog box has a title bar with a close button (X) and contains the text "Gửi email thành công!" (Email sent successfully!) and an "OK" button. The background form is slightly dimmed.



```
//Xử lý khi nhấn nút gửi
1 reference
private void bt_send_Click(object sender, EventArgs e)
{
    var senderEmail = tb_sender.Text;
    var receiverEmail = tb_rcver.Text;

    if (!IsValidEmail(senderEmail) || !IsValidEmail(receiverEmail))
    {
        MessageBox.Show("Địa chỉ email không hợp lệ.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    try
    {
        using (var client = new MailKit.Net.Smtp.SmtpClient())
        {
            client.Connect(SMTPHostName, SMTPHostPort, true);
            client.Authenticate(username, password);

            var message = new MimeMessage();
            message.From.Add(new MailboxAddress("", tb_sender.Text));
            message.To.Add(new MailboxAddress("", tb_rcver.Text));
            message.Subject = tb_subject.Text;

            //Tạo nội dung email
            var body = isHTML.Checked ? new TextPart("html") { Text = rtb_body.Text } : new TextPart("plain") { Text = rtb_body.Text };
            var multipart = new Multipart("mixed");
            multipart.Add(body);

            //Thêm tệp đính kèm vào email
            foreach (var path in attachmentPaths)
            {
                var attachment = new MimePart()
                {
                    Content = new MimeContent(File.OpenRead(path)),
                    ContentDisposition = new ContentDisposition(ContentDisposition.Attachment),
                    ContentTransferEncoding = ContentEncoding.Base64,
                    FileName = Path.GetFileName(path)
                };
                multipart.Add(attachment);
            }

            message.Body = multipart; //Gán nội dung email

            client.Send(message); //Gửi email
            client.Disconnect(true); //Gửi email

            MessageBox.Show("Gửi email thành công!");
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Gửi email thất bại. Lỗi: {ex.Message}");
    }
}
```

```
//Hàm kiểm tra tính hợp lệ của địa chỉ email
2 references
private bool IsValidEmail(string email)
{
    try
    {
        var addr = MailboxAddress.Parse(email);
        return true;
    }
    catch
    {
        return false;
    }
}

//Thêm tệp đính kèm
1 reference
private void AddAttachment(string path)
{
    attachmentPaths.Add(path);
    rtb_attach.Text = string.Join("\n", attachmentPaths);
}
```

```
1 reference
private void bt_browse_Click(object sender, EventArgs e)
{
    using (OpenFileDialog ofd = new OpenFileDialog())
    {
        ofd.Multiselect = true; // Cho phép chọn nhiều tập tin
        if (ofd.ShowDialog() == DialogResult.OK)
        {
            foreach (string path in ofd.FileNames)
            {
                AddAttachment(path);
            }
        }
    }
}
```