****

|  |  |
| --- | --- |
| **Họ và tên** | **MSSV** |
| Nguyễn Thanh Tùng | 23521744 |
| Nguyễn Quốc Việt | 23521782 |
| Đỗ Đức Minh Triết | 23521650 |

GVHD: **Phan Trung Phát**

Lớp: **NT106.P13.1**

5

**Sending & Receiving Email in C#**

**ĐÁNH GIÁ KHÁC (\*):**

|  |  |
| --- | --- |
| **Nội dung** | **Kết quả** |
| Tổng thời gian thực hiện bài thực hành trung bình (1) | 5 ngày |
| Link Video thực hiện (2)  *(nếu có)* | [Lab4](https://drive.google.com/file/d/1P9_SvVOBzXJ1M3tu4pXBhp8WgpAV1o8R/view?usp=sharing) [Lab5](https://drive.google.com/file/d/1yZ_XAgX9MwIlMud9QO0EVcNIHbRWQUIL/view?usp=sharing) [Lab6](https://drive.google.com/file/d/1XWbU7d843gy9n3XULIYS8SSp4rI-M4jL/view?usp=sharing) |
| Ý kiến (3) *(nếu có)*  + Khó khăn  + Đề xuất … |  |
| Điểm tự đánh giá (4) | 8.5/10 |
| (\*): phần (1) và (4) bắt buộc thực hiện. | |

# Câu hỏi 1

## 1. Minh chứng:

Nội dung.

A screenshot of a computer

Description automatically generated

Hình 1. Giao diện Bài 4

A screen shot of a computer program

Description automatically generated

Hình 2. Class Movie và localPath mặc định

A computer screen shot of a program code

Description automatically generatedA computer screen shot of a program

Description automatically generated

Hình 3. Function LoadMovieData();

A screen shot of a computer

Description automatically generated

Hình 4. Nội dung comboBox “chọn Phim” và class MovieStatistics

A screen shot of a computer program

Description automatically generated

A computer screen shot of a program code

Description automatically generated

Hình 5. Nội dung Button “Đặt vé”

A screen shot of a computer program

Description automatically generated

A computer screen with many colorful text

Description automatically generated

Hình 6. Nội dung Button “Xác nhận”

A screen shot of a movie title

Description automatically generated

Hình 7. Giao diện UserControl

A screen shot of a computer program

Description automatically generated

Hình 8. Nội dung của UserControl

## 2. Giải thích:

Khởi tạo class **Movie**/ List<**Movie**>movies:

- **Nội dung**: Khởi tạo class Movie với các thuộc tính **TenPhim**, **Url**, **Poster,** sau đó khởi tạo **List<Movie>** để lưu trữ thông tin phim.

String **localPath**:

- **Nội dung**: Đường đẫn thư mục để lưu Poster khi draw về từ trang web.

Hàm **LoadMovieData()**:

- **Nội dung**: Phương thức **bất đồng bộ (async)** trong C# để tải và hiển thị dữ liệu phim từ trang web **https://betacinemas.vn/phim.htm**. Sử dụng các thư viện như **HtmlAgilityPack** để phân tích **cú pháp HTML**. Đồng thời, tải về **hình ảnh** phim và hiển thị chúng trên một giao diện **WinForms** có **ListBox** và **FlowLayoutPanel**- **Trình bày**:

* **Url**: “<https://betacinemas.vn/phim.htm> là website dùng để draw thông tin về máy.
* var **Web**: Khởi tạo đối tượng **HtmlWeb** từ thư viện **HtmlAgilityPack** để tải và phân tích nội dung **HTML**.
* var **doc**: Tải nội dung trang web một cách không đồng bộ để tránh chặn luồng chính.
* var **movieNodes**: Chọn tất cả các phần tử **HTML** chứa dữ liệu phim bằng **XPath**. Lớp cụ thể **col-lg-4** được cho là được sử dụng cho các vùng chứa phim.
* int **movieCount**: Đếm số lượng phim.
* var **titleNode**: Tìm tiêu đề và liên kết của phim trong nút hiện tại bằng XPath.
* **Trích xuất nội dun**g văn bản của **tiêu đề phim** và cắt bớt khoảng trắng thừa.
* **Truy xuất URL** liên quan cho phim và kết hợp với URL cơ sở để tạo thành URL đầy đủ.
* var **imageNode:** Lấy đường dẫn hình ảnh từ thẻ **<img>** bên trong phần tử **<div>** có class **'pi-img-wrapper'**.
* Trích xuất thuộc tính **src** của node hình ảnh để lấy **URL.**
* **sanitizedTitle**: Xử lý tên phim để làm tên **file** hợp lệ (tìm tất cả các ký tự không phải là chữ cái **(in hoa hoặc thường)** hoặc chữ số).
* **movies.Add(movie)**: thêm phim vừa trích xuất vào collection.
* Thêm các tên phim vào **comboBox**.
* Khởi tạo **UserControl** cho mỗi phim.
* Kiểm tra trong thư mục đã có **poster phim** chưa, nếu chưa thì thực hiện **Download** sau đó load ảnh lên **PictureBox** của **UserControl**.
* **flowLayoutPanel1.Controls.Add(movieItem)**: Thêm mục phim vào **FlowLayoutPanel**.

**Dictionary**<string, MovieStatistics> **movieStatistics**:

- **Nội dung**: Dùng để chứa thông tin mỗi phim được đặt vé (tên phim, ảnh, doanh thu).

**List**<string> **selectedSeats**:

- **Nội dung**: Dùng để lưu thông tin mỗi ghế đã chọn của một phim.

Button **datVe**:

- **Nội dung**: Tính toán giá vé và xuất ra thông tin mua vé.

- **Trình bày**:

* Kiểm tra thông tin người dùng nhập vào có rỗng hay không.
* **Khởi tạo một Key** tượng trưng cho một phim nếu phim đó **chưa được đặt**.
* Set giá trị **true** cho các ghế của **Key** được mua.
* **Tính toán giá vé** dựa trên vị trí chỗ ngồi.
* **In ra thông tin** khách hàng và thông tin đặt vé.

Button **Confirm**:

- **Nội dung**: Xác nhận đặt vé và gửi mail tới email người mua.

- **Trình bày**:

* Khởi tạo đối tượng **MailMessage** với **mail gửi** mặc định [23521744@gm.uit.edu.vn](mailto:23521744@gm.uit.edu.vn).
* **Mail đến** được lấy từ input người dùng.
* **Mail.Subject**: Chủ đề của mail.
* String **htmlBody**: Tạo nội dung mail HTML.
* Lấy danh sách ghế đã chọn.
* Thêm **thông tin** yêu cầu vào **mail** (Họ tên khách hàng, tên phim, số ghế, tổng tiền).
* Đính kèm file ảnh.
* Cấu hình **SMTP client** port mặc định của gmail là **587**.
* Gửi mail đi và nhận thông báo.

Lớp **MovieItemControl** ở Form UserControl:  
- **Nội dung**: Là một **UserControl tùy chỉnh** trong C#. Được thiết kế để **hiển thị thông tin về các bộ phim** trong giao diện người dùng của ứng dụng Windows Forms. Nó bao gồm các thuộc tính như **tiêu đề phim (Title)**, **đường dẫn phim (Link)** và **hình ảnh poster (PosterImage)**, cùng với chức năng cho phép người dùng **mở một trang web** khi nhấn vào đường dẫn.

Thuộc tính **Title**:

- **Nội dung**: Đây là thuộc tính công khai dùng để **lấy và đặt nội dung của MovieTitle**, một Label trên UserControl để hiển thị **tiêu đề phim.**

**- Trình bày:**

* **Getter (get)**: Trả về nội dung của **MovieTitle**.
* **Setter (set)**: Đặt nội dung của **MovieTitle** thành giá trị được cung cấp.

Thuộc tính **Link**:

- **Nội dung**: Thuộc tính này dùng để **lấy và đặt đường dẫn phim**. Nó liên kết với **MovieURI**, có thể là một **Label** hoặc **LinkLabel.**

**- Trình bày:**

* **Getter (get)**: Trả về nội dung của **MovieURI**.
* **Setter (set)**: Đặt nội dung của **MovieURI** thành giá trị được cung cấp.

Thuộc tính **PosterImage:**

**- Nội dung:** Thuộc tính này dùng để **lấy và đặt hình ảnh poster** của phim. Nó liên kết với **Poster**, một **PictureBox** trên **UserControl**.

- **Trình bày**:

* **Getter (get)**: Trả về hình ảnh hiện tại của **Poster**.
* **Setter (set)**: Đặt hình ảnh cho **Poster** từ giá trị được cung cấp.

# Câu hỏi 5

## Minh chứng:

A screenshot of a computer

Description automatically generated

Hình 9 Form Invite

A screenshot of a computer

Description automatically generated

Hình 10 Form Email Setting

A screenshot of a computer

Description automatically generated

Hình 11 Code form Invite(1)

A screenshot of a computer

Description automatically generated

Hình 12 Code form Invite(2)  
A screenshot of a computer

Description automatically generated

Hình 13 Code form Invite(3)  
A screenshot of a computer

Description automatically generated

Hình 14 Code form Invite(4)

A screenshot of a computer

Description automatically generated

Hình 15 Code form Invite(5)

A screenshot of a computer

Description automatically generated

Hình 16 Code form Email Setting(1)  
A screenshot of a computer

Description automatically generated

Hình 17 Code form Email Setting(2)

A screenshot of a computer

Description automatically generated

Hình 18 Code form Email Setting(2)

## 2. Giải thích:

## Form Email Setting:

* **Email\_Setting (Constructor)**:
  + Khởi tạo form **Email\_Setting**.
  + Gọi **InitializeComponent** để thiết lập giao diện.
* **btn\_TestConnection\_Click**:
  + Kiểm tra kết nối SMTP dựa trên thông tin nhập từ giao diện (SMTP server, port, username, password).
  + Gửi một email test đến chính địa chỉ email nhập vào để kiểm tra xem SMTP có hoạt động chính xác không.
  + Hiển thị thông báo thành công hoặc lỗi tùy theo kết quả.
  + Bắt và xử lý các lỗi có thể xảy ra như lỗi SMTP, định dạng hoặc lỗi khác.
* **btn\_SaveExit\_Click**:
  + Thu thập thông tin cấu hình (IMAP/SMTP server, port, username, mật khẩu, trạng thái SSL).
  + Lưu thông tin này vào tệp JSON (email\_config.json) để dùng cho các chức năng khác.
  + Hiển thị thông báo lưu thành công hoặc lỗi nếu lưu thất bại.
  + Đóng form sau khi thực hiện.
* **Email\_Setting\_Load**:
* Gán các giá trị mặc định cho ComboBox cmb\_IMAPPort và cmb\_SMTPPort, gồm các cổng phổ biến:
  + IMAP: 143 (không SSL) và 993 (SSL).
  + SMTP: 25 (không SSL), 465 (SSL), và 587 (TLS).

## Form Invite:

* **Invite\_Load**:
  + Được gọi khi form được tải.
  + Đọc cấu hình email từ tệp JSON (email\_config.json) để thiết lập thông tin gửi email.
  + Kiểm tra và hiển thị thông tin món ăn (tên, giá, địa chỉ, người đóng góp, hình ảnh) trên giao diện form.
* **btn\_Add\_Click**:
  + Thêm địa chỉ email từ **txb\_Email** vào danh sách email **richTextBox1**.
  + Kiểm tra nếu người dùng chưa nhập email, thông báo yêu cầu nhập.
* **SendEmail**:
  + Gửi email đến một địa chỉ cụ thể.
  + Nội dung email chứa thông tin món ăn (tên, giá, địa chỉ, người đóng góp) và một liên kết xác nhận lời mời.
  + Sử dụng cấu hình SMTP từ tệp JSON để gửi email với nội dung HTML.
* **btn\_Review\_Click**:
  + Hiển thị nội dung email (món ăn, giá, địa chỉ, danh sách email) để người dùng xem lại trước khi gửi.
  + Nếu người dùng xác nhận (nhấn OK), sẽ tự động kích hoạt nút gửi email **btn\_Send**.
* **btn\_Send\_Click**:
  + Gửi email đến toàn bộ danh sách địa chỉ email trong **richTextBox1**.
  + Sử dụng SendEmail để gửi từng email.

# Câu hỏi 3

## 1. Minh chứng:

## :A computer screen shot of a program Description automatically generated

Hình 19. Code form chính bài 6 (1)

A computer screen shot of a program

Description automatically generated

Hình 20. Code form chính bài 6 (2)

A computer screen shot of a program

Description automatically generated

Hình 21. Code form chính bài 6 (3)

A computer screen shot of a program

Description automatically generated

Hình 22. Code form chính bài 6 (4)

A screenshot of a computer

Description automatically generated

Hình 23. Form chính bài 6

A computer screen shot of a program

Description automatically generated

Hình 24. Code form SendEmail (1)

A computer screen shot of a program

Description automatically generated

Hình 25. Code form SendEmail (2)

A screenshot of a computer

Description automatically generated

Hình 26. Form SendEmail

A computer screen shot of a program

Description automatically generated

Hình 27. Code form DisplayEmail

A screenshot of a computer

Description automatically generated

Hình 28. Form DisplayEmail

## 2. Giải thích:

## Form chính bài 6:

* Khởi tạo form:
* Thêm các cột số thứ tự email, Subject, From, Date cho **ListView** hiển thị danh sách email.
* Gắn sự kiện **this.FormClosing** với phương thức xử lý sự kiện **Bai6\_FormClosing.**
* Gắn sự kiện **MouseDoubleClick** của **ListView** với phương thức xử lý sự kiện **lvMails\_DoubleClick**.
* **FullRowSelect** của **ListView** bằng true để có thể chọn email trong **ListView**.
* Khởi tạo một **ImapClient**.
* **Bai6\_FormClosing**: thực hiện nhấn nút đăng xuất khi đóng form.
* Khi nhấn nút **“Đăng nhập”** phương thức **btDangNhap\_Click** thực hiện:
* Hiện **progressBar** để người dùng theo dõi tiến trình đăng nhập.
* Lấy các thông tin cần thiết như Email đăng nhập, mật khẩu, địa chỉ máy chủ IMAP, cổng kết nối từ giao diện form mà người dùng nhập vào.
* Tiến hành kết nối tới máy chủ IMAP với các thông tin cần thiết được lấy ở trên thông qua **ImapClient** được khởi tạo với form.
* Inbox: Truy cập hộp thư đến của người dùng. Mở hộp thư ở chế độ chỉ đọc để lấy email mà không làm thay đổi trạng thái email (ví dụ: không đánh dấu email là đã đọc).
* Tải danh sách email: Duyệt ngược từ email mới nhất (ở cuối danh sách) cho đến tối đa 40 email hoặc tất cả email nếu ít hơn 40 thông qua vòng lặp. **inbox.GetMessage(i)** để lấy chi tiết từng email, bao gồm chủ đề, người gửi, ngày giờ.
* Hiển thị email lên **ListView**: Các thông tin chi tiết của từng email được thêm vào các cột thông tin tương ứng của ListView thông qua biến message và item. Với cột số thứ tự email là biến count.
* Cập nhật giao diện sau khi đăng nhập: Nút **“Đăng Nhập”** được ẩn đi và các nút **“Đăng Xuất”**, **“Refresh”**, **“Gửi Mail”** được hiện lên là các chức năng có được sau khi đăng nhập thành công. Các trường thông tin người dùng nhập vào không thể chỉnh sửa sau khi đăng nhập thành công. Ẩn thanh trạng thái.
* Hiển thị thông báo: Một **MessageBox** thông báo đăng nhập thành công và thông báo lỗi đăng nhập.
* Khi nhấn nút “**Đăng Xuất**” phương thức **btDangXuat\_Click** thực hiện:
* Kiểm tra đối tượng **ImapClient** có khác null không, nếu có tiến thành ngắt kết nối client, thu hồi tài nguyên và gán null cho client.
* Cập nhật giao diện sau khi đăng nhập: Ẩn các nút **“Đăng Xuất”**, **“Gửi Email”**, **“Refresh”**, hiện nút **“Đăng Nhập”**, người dùng có thể chỉnh sửa các trường thông tin, **ListView** xóa các thông tin email.
* Khi nhấn nút “**Refressh**” phương thức **btDangXuat\_Click** thực hiện:
* Xóa danh sách email hiện tại: **lvMails.Items.Clear()** đảm bảo làm trống danh sách email trước khi tải lại.
* Truy cập hộp thư và tải danh sách email tương tự khi nhấn nút **“Đăng Nhập”**.
* Hiển thị thông báo hoàn tất: **MessageBox** thông báo khi danh sách email được làm mới xong.
* Khi nhấn nút **"Gửi Mail"** phương thức **btGuiMail\_Click** thực hiện:
* Tạo một form mới (**SendEmail**) để người dùng soạn và gửi email.
* Sử dụng phương thức **Authenticate** của form **SendEmail** để gửi thông tin về tài khoản và máy chủ SMTP, tài khoản và mật khẩu người dùng (cung cấp bởi người dùng thông qua các trường nhập trên giao diện chính) đến form **SendEmail**.
* Sau khi gửi thông tin, form gửi email sẽ được mở bằng **ShowDialog()**.
* Khi nhấn đúp chuột vào một email trong danh sách phương thức **lvMails\_DoubleClick** thực hiện:
* **SelectedItems.Count > 0** kiểm tra xem có email nào được chọn hay không.
* Lấy chỉ số của email được chọn trong danh sách **ListView** thông qua **SelectedItems[0].Index**.
* Do danh sách được hiển thị theo thứ tự mới nhất đến cũ nhất, sử dụng **inbox.Count - 1 - selectedIndex** để khớp với vị trí thực trong hộp thư.

Sử dụng **inbox.GetMessage(index)** để tải email với các thông tin như: người gửi, người nhận, subject, và nội dung (HTML hoặc văn bản thuần).

* Truyền thông tin email (người gửi, người nhận, chủ đề, nội dung) vào form **DisplayEmail** qua phương thức **display\_email** để hiển thị email chi tiết.
* Xử lý lỗi: Nếu không thể mở email, hiển thị thông báo lỗi với nội dung chi tiết qua **MessageBox**.

## Form SendEmail:

* Khởi tạo form với các biến lưu trữ mật khẩu, địa chỉ máy chủ SMTP và số cổng kết nối.
* Phương thức **Authenticate** nhận các thông tin cần thiết cho việc kết nối SMTP từ form chính. Tài khoản người dùng được hiển thị ra **tbFrom**, các thông tin còn lại được gán vào biến lưu trữ tương ứng.
* Khi nhấn nút **“Browse”** phương thức **btBrowse\_Click** thực hiện mở hộp thoại **OpenFileDialog** để người dùng chọn tệp bất kỳ. Hiển thị đường dẫn tệp đã chọn vào **tbAttachment**.
* Khi nhấn nút **“Send”** phương thức **btSend\_Click** thực hiện:
* Lấy dữ liệu từ các trường trên form: Địa chỉ người gửi (**tbFrom**), người nhận (**tbTo**), tiêu đề (**tbSubject**), nội dung email (**rtbBody**), đường dẫn tệp đính kèm (**tbAttachment**), và kiểm tra định dạng HTML (**cbHTML**).
* Kiểm tra tệp đính kèm: Nếu đường dẫn tệp không rỗng và tệp tồn tại, tệp sẽ được thêm vào email. Nếu tệp lớn hơn 20MB, hiển thị thông báo lỗi và hủy thao tác gửi email.
* Cấu hình kết nối SMTP: Tạo một đối tượng **SmtpClient** với địa chỉ máy chủ và cổng SMTP đã thiết lập. Sử dụng xác thực **NetworkCredential** với email và mật khẩu của người gửi. Bật mã hóa SSL để bảo mật kết nối.
* Gửi email: Tạo một đối tượng **MailMessage** để chứa các thông tin email, bao gồm địa chỉ gửi, nhận, tiêu đề, nội dung, và tệp đính kèm. Sử dụng **smtpClient.Send** để gửi email.
* **MessageBox** thông báo sẽ hiện ra nếu gửi email thành công hoặc thất bại.

## Form DisplayEmail:

* Khởi tạo form với sự kiện **this.Resize** gắn với phương thước xử lý sự kiện **DisplayEmail\_Resize**.
* Phương thức **DisplayEmail\_Resize** thay đổi kích thước của **webBrowserEmail** tương ứng với kích thước form hiện tại theo tùy chỉnh của người dùng.
* Phương thức **display\_email** nhận các thông tin cần thiết từ form chính và thực hiện:
* Hiển thị thông tin người gửi và người nhận, tiêu đề qua các Label tương ứng (**lbFrom**, **lbTo**, **lbSubject**).

Đặt tiêu đề email làm tên form và đặt nội dung email vào **webBrowserEmail**.

**YÊU CẦU CHUNG**

1. Đánh giá

* Chuẩn bị tốt các yêu cầu đặt ra trong bài thực hành.
* Sinh viên hiểu và tự thực hiện được bài thực hành, trả lời đầy đủ các yêu cầu đặt ra.
* Nộp báo cáo kết quả chi tiết những đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả *(nếu có)*; giải thích cho quan sát *(nếu có)*.
* Sinh viên báo cáo kết quả thực hiện và nộp bài.

1. Báo cáo

* Nộp file .docx. Tập trung vào nội dung, giải thích.
* Nội dung trình bày bằng Font chữ Cambria hoặc Times New Roman (*tuy nhiên, phải chuyển đổi hết báo cáo này sang 1 font chữ thống nhất*) – cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
* Đặt tên theo định dạng: Mã lớp-LabX\_MSSV1\_MSSV2. (trong đó X là Thứ tự buổi Thực hành).

Ví dụ: NT106.O21.1-Lab01\_25520001\_25520002

* Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
* Không đặt tên đúng định dạng – yêu cầu, sẽ KHÔNG chấm điểm bài thực hành.
* Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.

**HẾT**