4

Lab

Truyền thông với Web Server trong C#

Working with Web Server in C#

Thực hành Lập trình mạng căn bản

GVHD: Nguyễn Xuân Hà

Lưu hành nội bộ

A. TỔNG QUAN

1. Mục tiêu

- Nắm được cách giao tiếp với Web Server trong C# và vận dụng để viết các ứng dụng gửi, nhận dữ liệu, truyền thông với Web Server.
- Làm quen và tìm hiểu cách tự xây dựng một website đơn giản với HTML.

2. Môi trường

- IDE Microsoft Visual Studio 2010 trở lên.

3. Liên quan

- Sinh viên cần nắm được các kiến thức nền tảng về lập trình. Các kiến thức này đã được giới thiệu trong các môn học trước và trong nội dung lý thuyết đã học do đó sẽ không được trình bày lại trong nội dung thực hành này.
- Tham khảo tài liệu (Mục E).

B. THỰC HÀNH

1. Giới thiệu về xây dựng các ứng dụng tương tác với Internet Server

Microsoft .NET Framework hỗ trợ khả năng phân lớp, mở rộng và hỗ trợ tích hợp các dịch vụ Internet nhanh chóng và dễ dàng vào các ứng dụng. Các ứng dụng mạng có thể xây dựng trên các giao thức được xây dựng sẵn để tương tác với các server.

Các kiến thức liên quan đến nội dung thực hành trong Lab 4, sinh viên cũng có thể tham khảo thêm chi tiết tại *https://docs.microsoft.com/en-us/dotnet/framework/network-programming/*, *tập trung vào nội dung* ***Requesting Data*** - *https://docs.microsoft.com/en-us/dotnet/framework/network-programming/requesting-data*

Ngoài ra, sinh viên có thể tham khảo từ kiến thức từ Nội dung Lý thuyết Chương 4 (Web Server).

2. Cách tạo một Website đơn giản chạy trên localhost

Bước 1: Mở một IDE hỗ trợ soạn thảo, viết mã nguồn HTML (Nodepad, Visual Studio,…), và tạo một file với đuôi là .html. Ví dụ: index.html

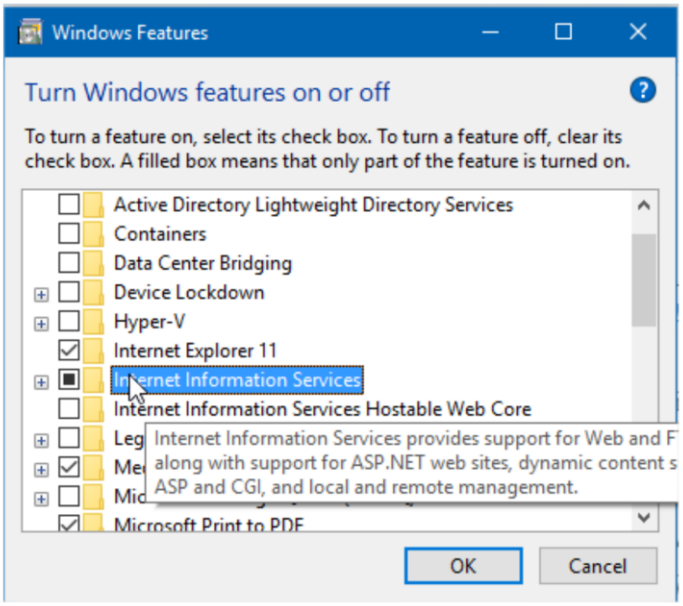
Bước 2: Viết mã nguồn để tạo một Website đơn giản. Tham khảo file bên dưới

Lưu ý: Sinh viên có thể tìm hiểu thêm về HTML tại: https://www.w3schools.com/ hoặc tự tạo một trang web khác tương đương bằng HTML, CSS.

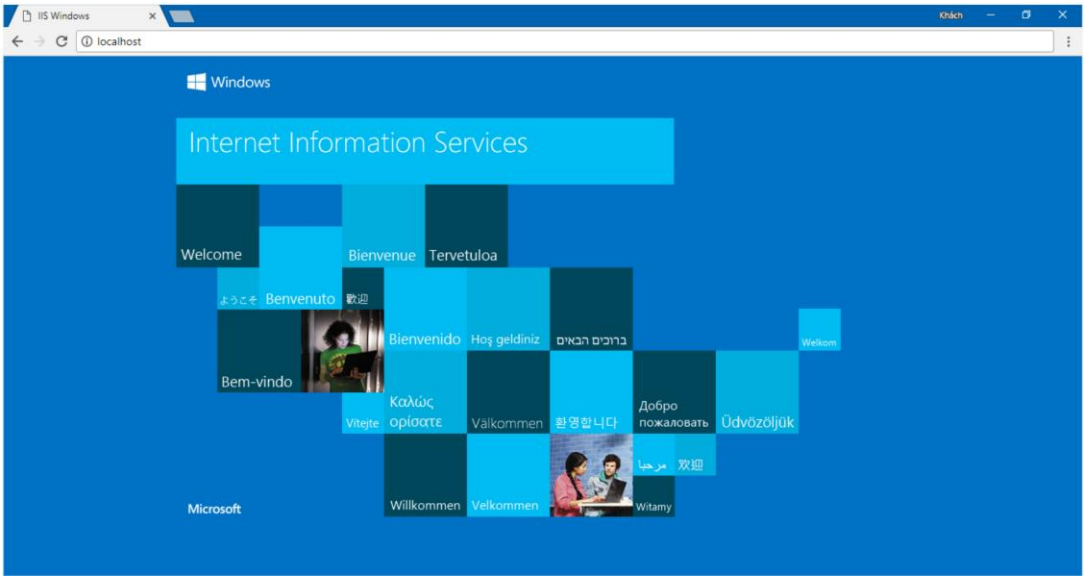
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>UIT - NT106 Lab 4 - Working with Web Server</title>
    <meta name="description" content="Lab4" />
    <meta name="keywords" content="Lab4"  />
</head>
<body class='pushmenu-push' id="page">
    <div class="dialog" style="width:60%;margin:10% auto;text-align:center;">
        <div class="dialogBox">
        <h4>Chào bạn!</h4>
        <h3>NT106 Lab 4 - Working with Web Server</h3>
       <img src="https://portal.uit.edu.vn/Styles/profi/images/logo186x150.png"
       alt="Trường Đại học Công nghệ Thông tin" role="presentation">
       <div style="color:blue;">Bạn đang làm bài Lab 4 của Môn học</div>
       <br/>
       <i>Chúc bạn hoàn thành bài Lab thành công.</i>
       </div>
    </div>
 </body>
```

Đoạn mã nguồn trên, khi chạy trên trình duyệt sẽ hiển thị dưới dạng như sau:

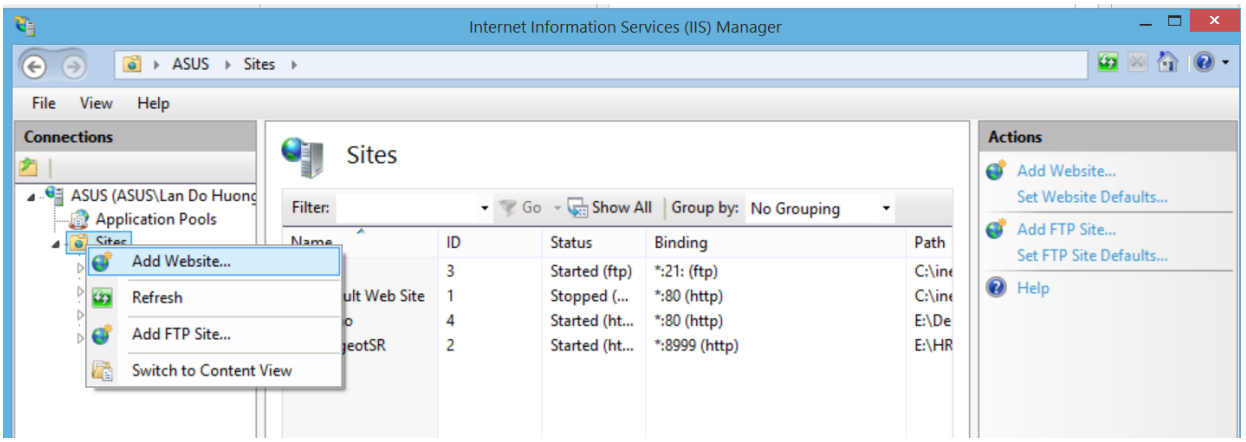
Bước 3: Bật dịch vụ **IIS** (dịch vụ để tạo máy chủ Web/FTP,...) trên **Windows 7/8/10** với các bước như sau: **Control Panel → Programs and features → Turn Windows Features on or off →** Chọn **Internet Information Service → OK.**

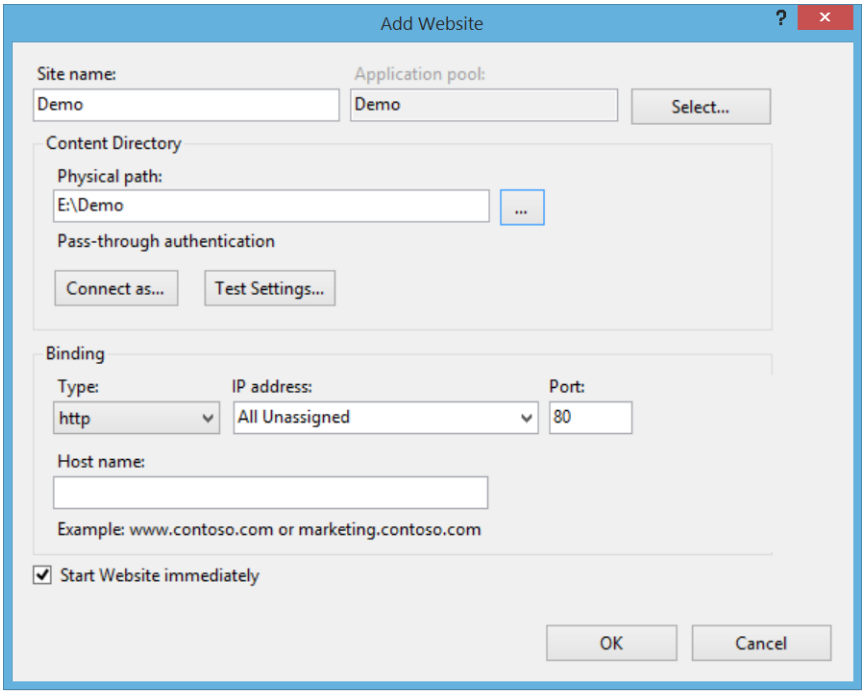


Bước 4: Sau khi bật dịch vụ IIS, kiểm tra dịch vụ đã hoạt động hay chưa bằng cách sử dụng một trình duyệt bất kỳ, và gõ **localhost**, nếu một trang Web mặc định của IIS hiện ra là thành công.

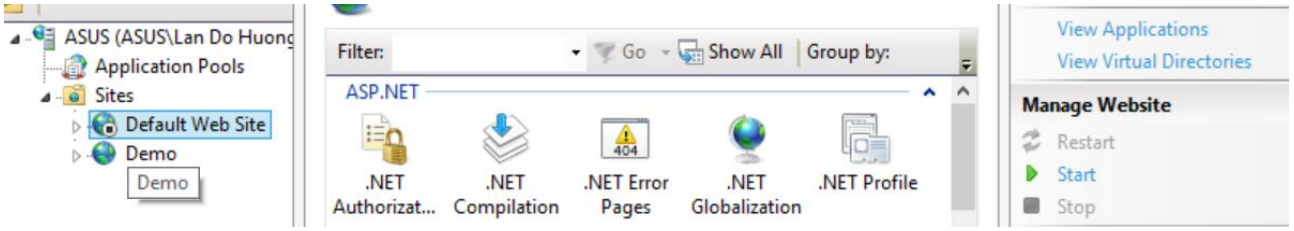


Bước 5: Lúc này ta đã có một máy chủ web đang hoạt động ở địa chỉ localhost. Mở IIS, để chạy một website trên địa chỉ này, nhấp chuột phải vào **Site** và **Add Website**. Với **Physical path** là tên folder chứa file html đã tạo ở Bước 1,2.





Lưu ý: Dừng chạy (Stop) trang mặc định của IIS và **Chạy (Start)** trang vừa khởi tạo để địa chỉ localhost chạy web site mới.



Bước 6: Truy cập vào trang web với đường dẫn http://localhost/index.html hoặc http://locahost. Truy cập thử trang web của sinh viên khác từ trình duyệt bằng các gõ URL như sau: http://A.B.C.D/index.html với A.B.C.D là địa chỉ IP của máy tính mà bạn mình sử dụng.

Lưu ý: Sinh viên và bạn mình phải sử dụng chung mạng Local (LAN).

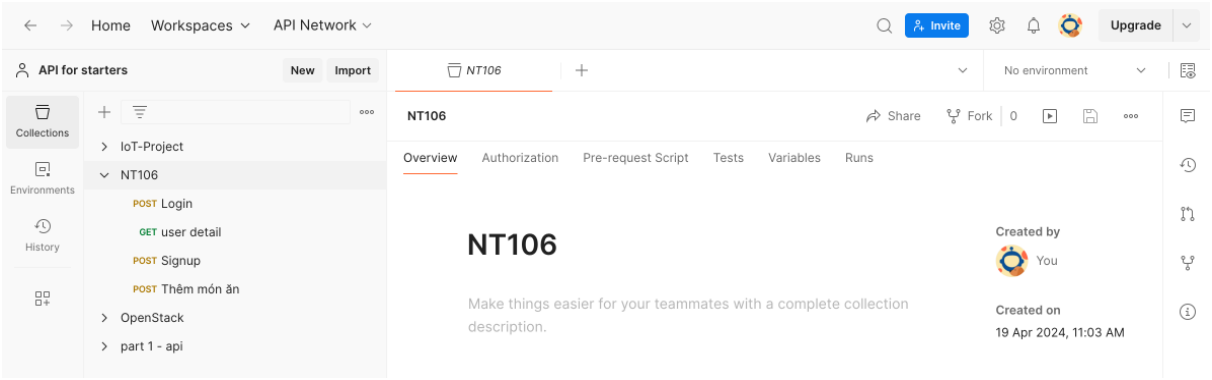
3. Giới thiệu Postman để kiểm tra API

API là cơ chế cho phép 2 "ứng dụng" giao tiếp với nhau bằng tập hợp các định nghĩa và giao thức. Ví dụ: hệ thống phần mềm của cơ quan thời tiết chứa dữ liệu về thời tiết hàng ngày. Ứng dụng thời tiết trên điện thoại của bạn sẽ "trò chuyện" với hệ thống này qua

API và hiển thị thông tin cập nhật về thời tiết hàng ngày trên điện thoại của bạn. Từ đó, nó có thể giúp việc trao đổi giữa các ứng dụng diễn ra thuận tiện hơn [1].

Web API là hệ thống API được sử dụng trong các hệ thống website. Hầu hết các website đều ứng dụng Web API nhằm cho phép bạn kết nối, lấy dữ liệu hoặc cập nhật cơ sở dữ liệu. Ví dụ: Bạn thiết kế chức năng đăng nhập thông qua Google, Facebook, Twitter, Github… Điều này có nghĩa là bạn đang gọi đến API của các ứng dụng này. Hoặc như các ứng dụng di động đều lấy dữ liệu thông qua API [2]. **REST API** là loại API phổ biến và linh hoạt nhất trên web hiện nay.

Postman hiện là một trong những công cụ phổ biến nhất được sử dụng trong việc thử nghiệm các API. Với Postman, ta có thể gọi REST API mà không cần viết dòng code nào. Postman hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, PATCH, DELETE…). Bên cạnh đó, nó còn cho phép lưu lại lịch sử các lần request, rất tiện cho việc sử dụng lại khi cần [3].

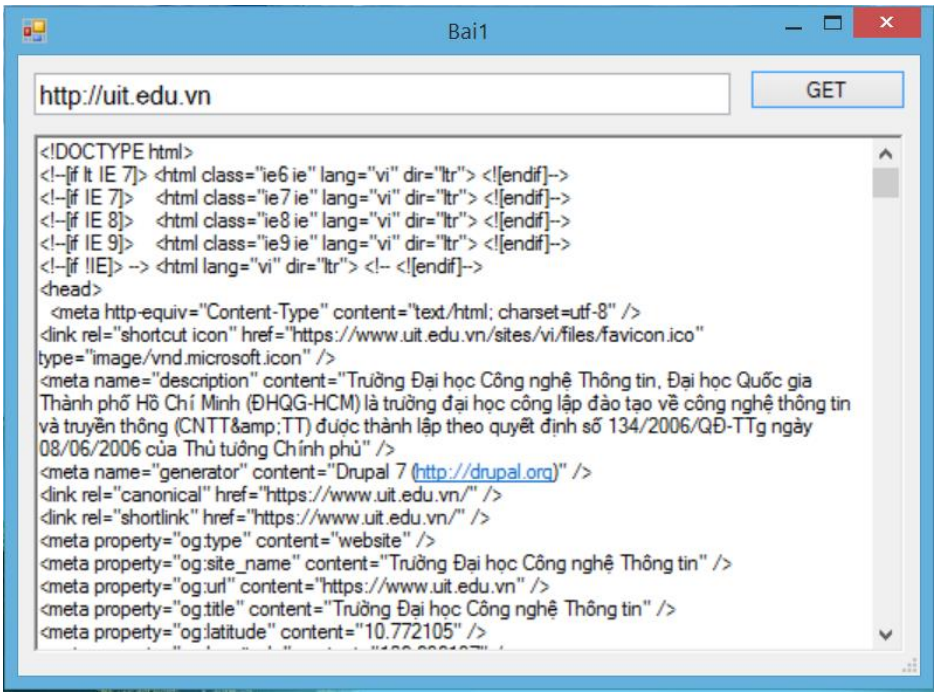


[1] https://aws.amazon.com/vi/what-is/api/
[2] https://topdev.vn/blog/api-la-gi/
[3] https://topdev.vn/blog/postman-la-gi/

4. Nội dung thực hành

Bài 1: Viết chương trình hiển thị nội dung HTML của một trang web bất kỳ:

Giao diện tham khảo:



Gợi ý: Sử dụng WebRequest và WebResponse:

- Khởi tạo WebRequest với đường link web bất kỳ.
- Dùng Stream để đón nhận dữ liệu từ WebResponse trả về.
- Tải nội dung trang web lưu vào biến String.
- Đọc dữ liệu và in ra màn hình.

Lưu ý: Khai báo sử dụng các thư viện cần thiết.

Code tham khảo: Hàm lấy dữ liệu sử dụng WebRequest:

```
private string getHTML(string szURL)
  {
          // Create a request for the URL.
          WebRequest request = WebRequest.Create(szUrl);
          // Get the response.
          WebResponse response = request.GetResponse();
          // Get the stream containing content returned by the server.
          Stream dataStream = response.GetResponseStream();
          // Open the stream using a StreamReader for easy access.
          StreamReader reader = new StreamReader(dataStream);
          // Read the content.
          string responseFromServer = reader.ReadToEnd();
          // Close the response.
          response.Close();
          return responseFromServer;
```

```
    }
```

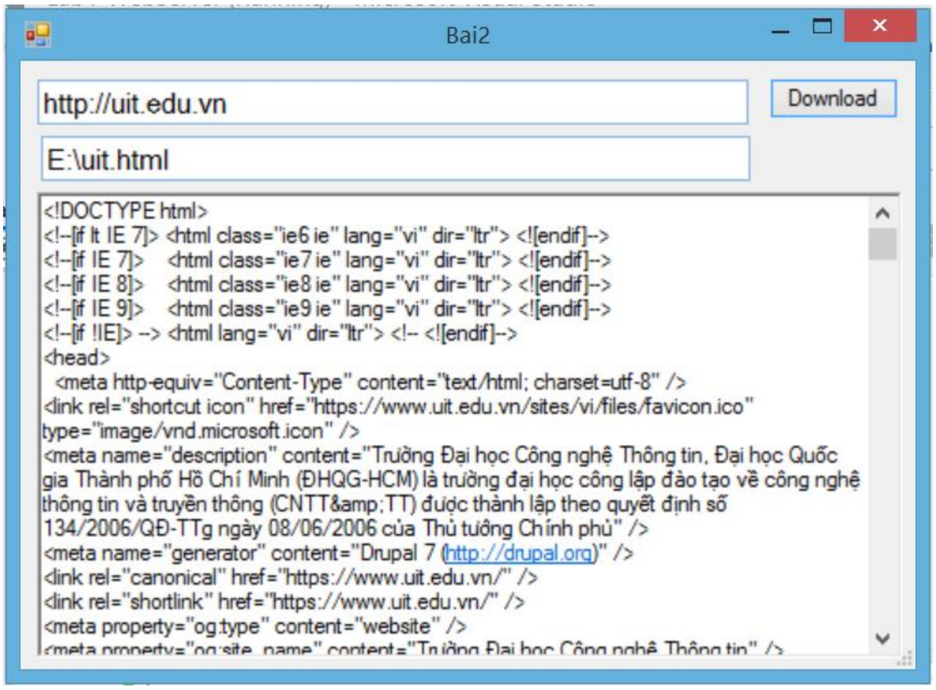
Sinh viên tự viết hàm bắt sự kiện nút **GET** và gọi hàm **getHTML**, hiển thị thông điệp từ hàm **getHTML** vào vùng hiển thị (giả sử dùng RichTextBox).

Tham khảo thêm tại:

https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-request-data-using-the-webrequest-class

Bài 2: Viết chương trình download nội dung trang web bất kỳ từ một địa chỉ URL bất kỳ và ghi thành file HTML, sau đó hiển thị nội dung trang web lên form.

Giao diện tham khảo:



Gợi ý: Sử dụng WebClient, và dùng phương thức DownloadFile của WebClient để lấy file về:

- Khởi tạo 1 WebClient.
- Sử dụng phương thức OpenRead để đọc nội dung trang web vào một Stream.
- Viết code xử lý sự kiện khi nhấn nút download, với 1 textbox chứa địa chỉ URL cần get nội dung, 1 textbox chứa đường dẫn lưu file chứa nội dung cần get về. Sử dụng thuộc tính DownloadFile của WebClient để get nội dung trang web về lưu vào file đã chỉ định ở trên.

```
WebClient myClient = new WebClient();
Stream response = myClient.OpenRead(url);
```

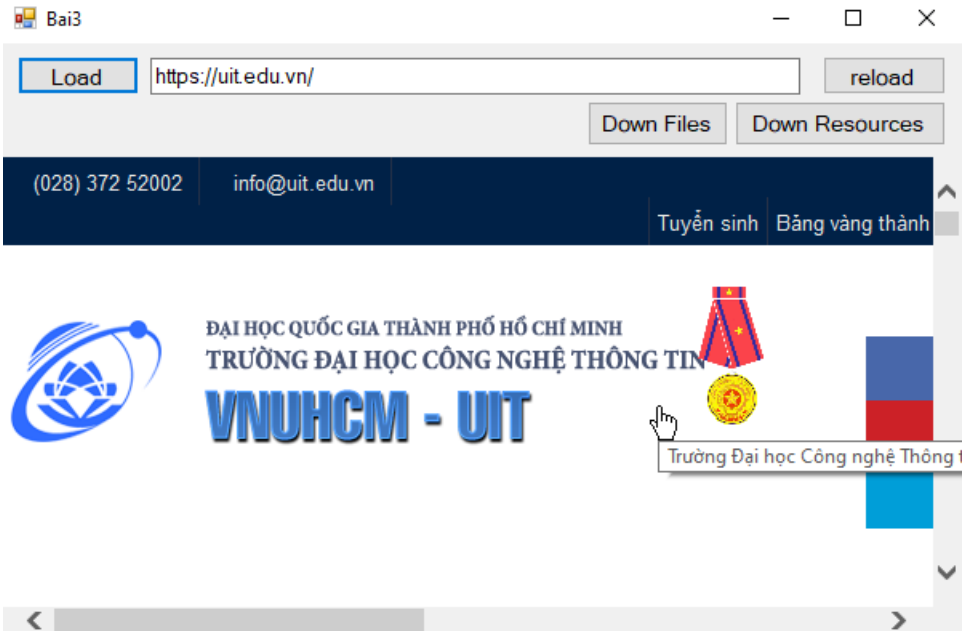
```
myClient.DownloadFile(url, fileurl);
```

Tham khảo thêm về WebClient tại:
https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-request-a-web-page-and-retrieve-the-results-as-a-stream

Bài 3: Viết chương trình hoạt động như một Web Browser cơ bản cho phép thực hiện các tính năng sau:

- Xem nội dung Website
- Download File html
- Download Resource của Website

Giao diện tham khảo:



Gợi ý:

- Sử dụng Control **WebView2** để hiển thị nội dung của website.
- Chức năng "Download Resource" cho phép tải tất cả hình của từ website (Sử dụng **HTMLAgilityPack** để trích xuất dữ liệu từ HTML file).
- Chức năng "View Source" bằng cách tạo một Form mới chứa mã nguồn HTML của website.
- Để sử dụng HTMLAgilityPack, sử dụng Nuget Packet để thêm vào Project.

Tham khảo cách thêm và sử dụng một packet tại:

https://docs.microsoft.com/en-us/nuget/quickstart/install-and-use-a-package-in-visual-studio

Tham khảo cách sử dụng Webview2 tại:

https://learn.microsoft.com/en-us/microsoft-edge/webview2/get-started/winforms

Tham khảo cách sử dụng Html Agility Pack tại:

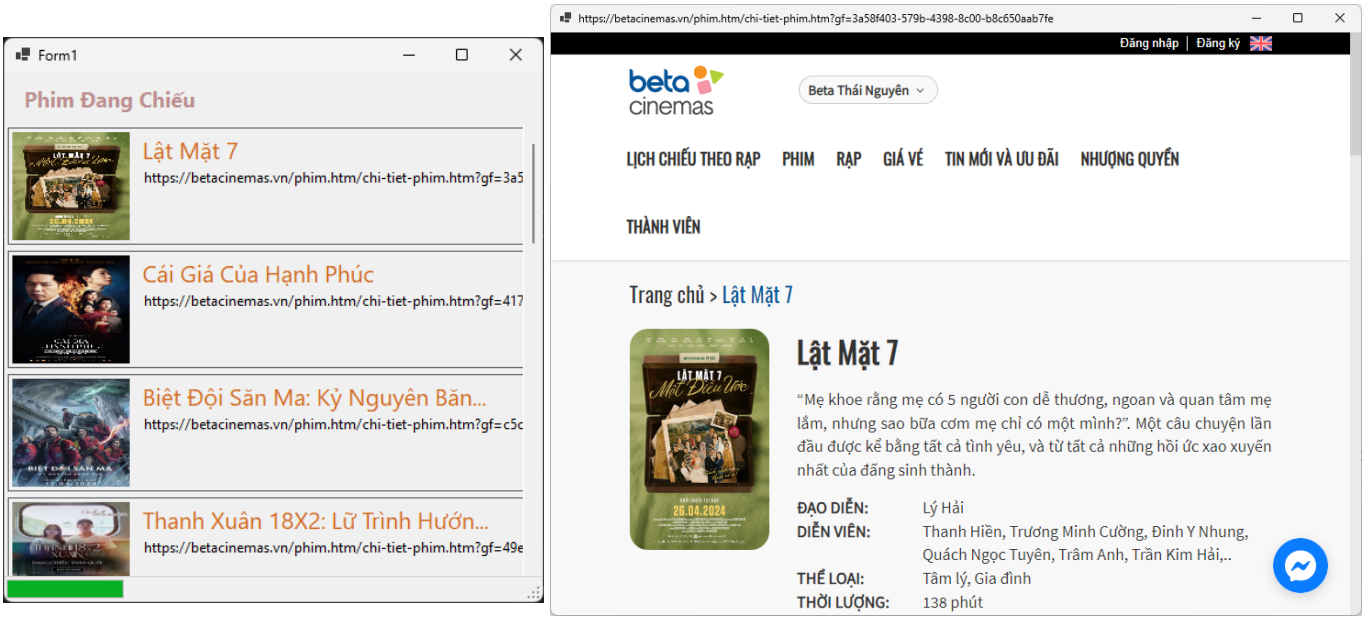
https://html-agility-pack.net/documentation

Bài 4: Quản lý phòng vé (phiên bản số 4). Lấy ý tưởng và kế thừa từ bài 5 - bài thực hành số 2. Viết một ứng dụng cho phép draw dữ liệu từ một website đặt vé xem phim. Dữ liệu tóm tắt về thông tin của bộ phim sẽ được lưu trữ dưới dạng JSON file. Hệ thống giúp hỗ trợ nhân viên đặt vé cho khách hàng, thông tin về khách hàng, vé và tiền thanh toán sẽ được thông báo ngay khi đặt vé.

Lưu ý:

- Website gợi ý: https://betacinemas.vn/phim.htm
- Thanh ProgressBar để hiển thị tiến trình trích xuất thông tin từ website.
- Nếu chọn vào banner của 1 bộ phim bất kỳ, truy cập đường dẫn đến bộ phim đó tại website gợi ý để hiện thị thông tin chi tiết của bộ phim.

Giao diện tham khảo:



Bài 5: HTTP POST - Viết chương trình cho phép đăng nhập vào ứng dụng Web thông qua API được cung cấp sẵn.

Hệ thống API được cung cấp sẵn tại: https://nt106.uitiot.vn/docs

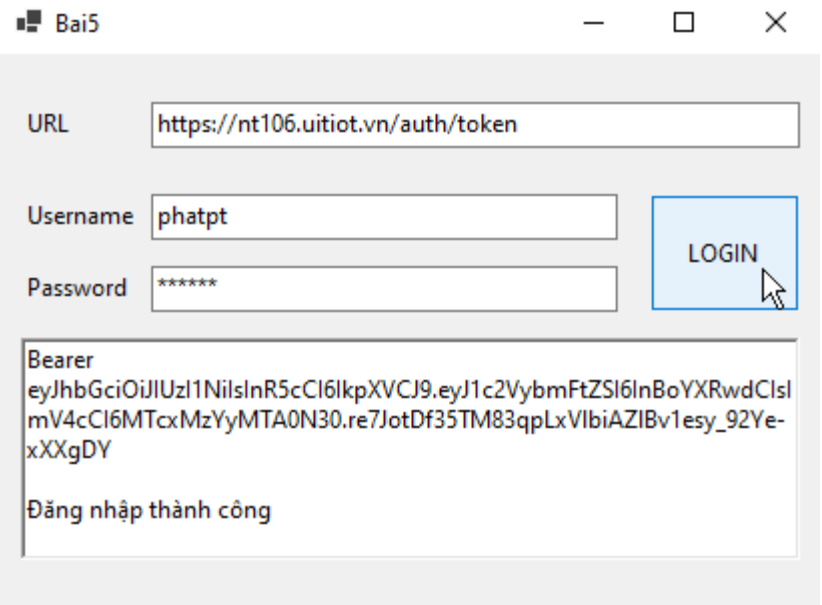
Gợi ý:

- Khởi tạo một kết nối HTTP POST tới địa chỉ: https://nt106.uitiot.vn/auth/token
- Truyền 2 tham số username và password trong form-data của Body HTTP POST.
- Tìm hiểu cách sử dụng dữ liệu JSON để truy xuất các trường thông tin trong JSON Object được trả về tại "access_token" và "token_type".
- In ra tình trạng của ứng dụng sau khi thực hiện một kết nối HTTP POST thông qua API:
    - Nếu đăng nhập thành công, in ra thông tin dạng:

    token_type access_token

    "Đăng nhập thành công"
    - Nếu đăng nhập không thành công, in ra thông tin trong trường "detail".

Giao diện tham khảo:



Tham khảo thêm tại:

https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-send-data-using-the-webrequest-class

The example code:

```
using System;
using System.Net.Http;
using System.Threading.Tasks;
using Newtonsoft.Json.Linq;

class Program
{
    static async Task Main(string[] args)
    {
```

```
        var url = "https://nt106.uitiot.vn/auth/token";
        var username = "phatpt"; // replace with your username
        var password = "123456"; // replace with your password

        using (var client = new HttpClient())
        {
            // Get the access token
            var content = new MultipartFormDataContent
            {
                { new StringContent(username), "username" },
                { new StringContent(password), "password" }
            };

            var response = await client.PostAsync(url, content);
            var responseString = await response.Content.ReadAsStringAsync();

            var responseObject = JObject.Parse(responseString);

            if (!response.IsSuccessStatusCode)
            {
                var detail = responseObject["detail"].ToString();
                Console.WriteLine($"Detail: {detail}");
                return;
            }

            var tokenType = responseObject["token_type"].ToString();
            var accessToken = responseObject["access_token"].ToString();

            Console.WriteLine($"Token Type: {tokenType}");
            Console.WriteLine($"Access Token: {accessToken}");

            // Use the access token to authenticate a GET request
            client.DefaultRequestHeaders.Authorization = new
System.Net.Http.Headers.AuthenticationHeaderValue("Bearer", accessToken);

            var getUserUrl = "https://nt106.uitiot.vn/api/v1/user/me";
            var getUserResponse = await client.GetAsync(getUserUrl);
            var getUserResponseString = await
getUserResponse.Content.ReadAsStringAsync();

            Console.WriteLine(getUserResponseString);
        }
    }
}
```

Bài 6: HTTP GET - Viết chương trình hiển thị thông tin người dùng hiện tại đang đăng nhập vào ứng dụng Web thông qua API được cung cấp sẵn.

Gợi ý:

- Khởi tạo một kết nối HTTP GET tới địa chỉ: https://nt106.uitiot.vn/api/v1/user/me

- Tìm hiểu phương thức xác thực giữa Client - Server với **JWT**.
- Thêm token đã thu được ở Bài 5 vào trường thông tin Authorization trong headers của HTTP GET.
- Hiển thị các thông tin cơ bản của user ra giao diện Form.

Tham khảo thêm tại:

https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-send-data-using-the-webrequest-class

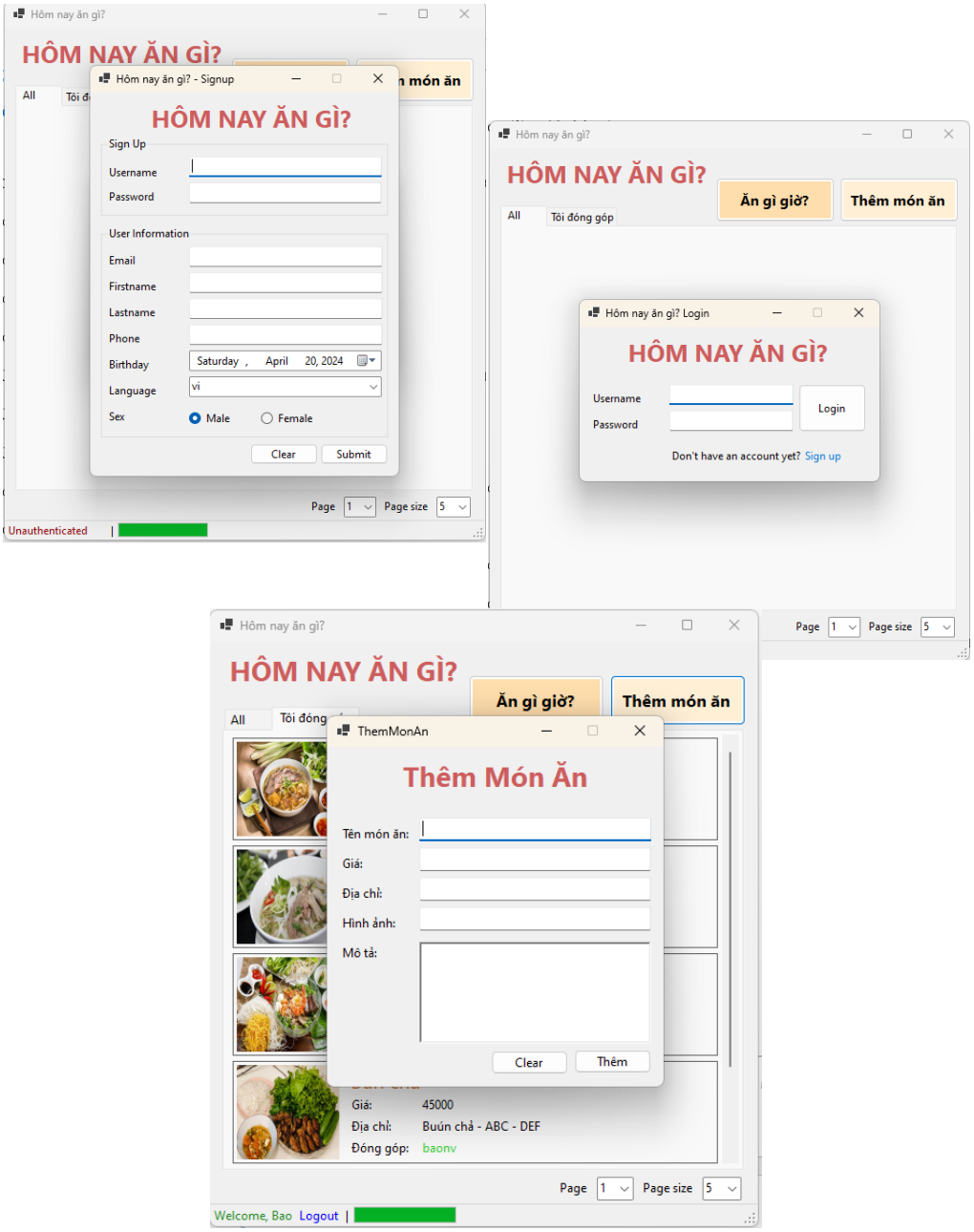
The example code:

```
client.DefaultRequestHeaders.Authorization = new
System.Net.Http.Headers.AuthenticationHeaderValue(tokenType, accessToken);

 var getUserUrl = "https://nt106.uitiot.vn/api/v1/user/me";
 var getUserResponse = await client.GetAsync(getUserUrl);
 var getUserResponseString = await
getUserResponse.Content.ReadAsStringAsync();

 Console.WriteLine(getUserResponseString);
```

Bài 7: Hôm nay ăn gì? (phiên bản số 4). Lấy ý tưởng và kế thừa từ bài 5 - bài thực hành số 3 và dựa trên hệ thống API đã cung cấp sẵn. Chức năng chính của ứng dụng vẫn dùng để ngẫu nhiên chọn ra 1 món ăn từ dữ liệu của bản thân hoặc dữ liệu của cả cộng đồng.

Lưu ý:

- Các tính năng cơ bản của một ứng dụng phải được đảm bảo:
  - Tạo tài khoản, đăng nhập:



  - Thêm 1 món ăn mới:

  - Xoá 1 món ăn.

- Chức năng hiển thị tất cả các món ăn có trên hệ thống, phân trang:

- Chức năng hiển thị các món ăn do bản thân tạo, phân trang.
- Chức năng chọn ngẫu nhiên một món ăn:

  - Ngẫu nhiên trong tất cả món ăn từ cả cộng đồng.
  - Ngẫu nhiên trong những món ăn bản thân đóng góp.

C. YÊU CẦU & NỘP BÀI

1. Yêu cầu

- Mỗi bài tập là 1 project và đặt trong cùng 1 solution.

- Mỗi project đều có form điều hướng để mở các form liên quan.
- Các giao diện ở trên chỉ mạng tính chất minh họa, sinh viên tiến hành thiết kế giao của riêng mình đảm bảo các tiêu chí: dễ nhìn, thể hiện hết được các yêu cầu cần thực hiện, đẹp.
- Code "sạch" [2], đặt tên biến rõ ràng.
- Nộp bài không đầy đủ; lỗi, không chạy được; nộp trễ; sao chép code bạn khác, nguồn có sẵn: *xử lý tùy theo mức độ.*

2. Đánh giá kết quả

- Sinh viên thực hành và nộp bài theo **Nhóm (Nhóm trưởng nộp)** tại website môn học theo thời gian quy định.
- Bài nộp bao gồm toàn bộ Source-code của các bài tập liên quan tại GitHub và trình bày báo cáo gồm Ảnh chụp màn hình kèm mô tả, giải thích các bước hoạt động, thực hiện của ứng dụng đã viết trong từng bài:

Toàn bộ project đặt vào 1 file nén (.zip) với tên theo quy tắc sau:

Mã lớp-LabX-MSSV1-MSSV2

Ví dụ: *NT106.O21.1-Lab04-25520001-25520002*

D. THAM KHẢO

[1] Microsoft (2018). C# Guide. [Online] Available at: https://docs.microsoft.com/en-us/dotnet/csharp/

[2] Martin, R. C. (2009). *Clean code: a handbook of agile software craftsmanship*. Pearson Education.

[3] Network Programming in the .NET Framework: https://docs.microsoft.com/en-us/dotnet/framework/network-programming/

HẾT