

Hướng Dẫn Thực Hành

Phát triển Web Kinh Doanh



Hướng dẫn:

- Bài tập thực hành được chia làm nhiều Module
- Mỗi Module được thiết kế cho thời lượng là 3 hoặc 5 tiết thực hành tại lớp với sự hướng dẫn của Giảng viên.
- Tùy theo số tiết phân bổ, mỗi tuần học có thể thực hiện nhiều Module.
- Sinh viên phải làm tất cả các bài tập trong các Module ở tuần tương ứng. Những sinh viên chưa hoàn tất phần bài tập tại lớp có trách nhiệm tự làm tiếp tục ở nhà.
- Các bài có dấu (*) là các bài tập nâng cao dành cho sinh viên khá giỏi.

Biên soạn và tổng hợp

TS. Trần Duy Thành

Mục Lục

Module 1: Tổng quan về Phát triển Web kinh doanh	6
Bài 1:History of the Internet	6
Bài 2:Server vs Client	7
Bài 3:HTTP vs HTTPS	7
Bài 4:Static Web and Dynamic Web	7
Bài 5:Hosting vs Domain	7
Bài 6:Server- Webserver-website.....	7
Bài 7: Structure of URL.....	7
Bài 8:Quy trình tạo lập một Website	7
Module 2 – Hypertext Markup Language (HTML)	8
Bài 9: Sử dụng Visual Studio để tạo Website	8
Bài 10: Cấu hình IIS Webserver.....	18
Bài 11: Cấu hình Hosting + Domain Free	28
Bài 12: Cấu hình NAT port cho Modem Internet ở nhà (*).....	29
Bài 13: Cấu trúc tập tin HTML	29
Bài 14: Resume	30
Bài 15: Beethoven	30
Bài 16: Euler.....	33
Bài 17: Chester.....	33
Bài 18: Resume bổ sung	34
Bài 19: Website Expo	35
Bài 20: Website Jaction	36
Bài 21: Website SFSF	37
Bài 22: Web Menu.....	38
Bài 23: Table	38
Bài 24: Table	39
Bài 25: Table Gargoyle	40
Bài 26: Table Dunston	41
Bài 27: Frame	41
Bài 28: Frameset	42
Bài 29: Frameset – Image maps	43
Bài 30: Frameset – Image maps	44
Bài 31: Frameset (*).....	44
Bài 32:Marquee, Multimedia	45
Module 3 – Cascading Style Sheets (CSS)	46
Bài 33: Internal Style	46
Bài 34: Inline Style.....	47
Bài 35: External Style	48



Bài 36: Box Menu.....	49
Bài 37: Menu cho Website	50
Bài 38: Bón mùa	52
Bài 39: Table with external CSS	53
Bài 40: Table with external CSS	54
Bài 41: Vietnam Airline Website – external CSS	54
Bài 42: Maxwell Scientific.....	55
Bài 43: StuffShop	56
Bài 44: M.Lee's.....	57
Bài 45: ScrapBooks.....	58
Bài 46: Mount Rainier News	58
Bài 47: Table with Style-trigger	59
Module 4 – Form và các Control trên Form	60
Bài 48: Register	60
Bài 49: Online Classifield	61
Bài 50: Travel Expense Report.....	62
Bài 51: Registration.....	63
Bài 52: Contact - GET	63
Bài 53: Sản phẩm-GET	64
Module 5 – Javascript	65
Bài 54: Hàm toán học.....	65
Bài 55: Giải phương trình bậc 2	65
Bài 56: Xử lý chuỗi	66
Bài 57: Xử lý sự kiện text change	66
Bài 58: Neonatal Feeding Study.....	67
Bài 59: North Pole Novel Ties.....	68
Bài 60: Menu	68
Bài 61: Color Picker	69
Module 6 – XML - Mô hình DOM và JSON	70
Bài 62: Table Mouse Trigger	70
Bài 63: Đăng ký thành viên.....	70
Bài 64: Dynamic Table	71
Bài 65: Dynamic List	72
Bài 66: XML-Dom Parser	73
Bài 67: XML – DOM Parser (*)	73
Bài 68: XML - AJAX	75
Bài 69: XML – AJAX (*).....	76
Bài 70: JSONObject vs JSONArray	76
Bài 71: Hiển thị thông tin User.....	77

Bài 72:Danh sách sản phẩm.....	77
Bài 73: Danh sách sản phẩm.....	78
Bài 74: Hiển thị bảng điểm- JSon-AJAX.....	79
Bài 75: Hiển thị Sách môn học - JsonArray	80
Module 7: AngularJS – Binding - Component	81
Bài 76 – Kiến trúc thành phần của Angular	81
Bài 77 – Cài đặt và lập trình Angular	81
Bài 78 - Binding.....	81
Bài 79 – Binding Property.....	82
Bài 80 – Binding Class	82
Bài 81 – Binding Style.....	83
Bài 82 – Binding Event.....	84
Bài 83 – Binding Two-Way	85
Bài 84 - Binding Two-Way, model for Quadratic Equation	85
Bài 85- Binding Two-Way, model for Lunar Year (*)	86
Module 8: AngularJS(tt) – Service - Routing	87
Bài 86- Json Object Model - Product	87
Bài 87- Json Array Model - Product.....	87
Bài 88- Json Array Model – Product Event (*).....	88
Bài 89- Json Array Model – Product - Catalog.....	92
Bài 90- Json Array Model – Product– Http Service(*).....	93
Bài 91- Json Array Model – Product– Http Service Handle Error (*)	95
Bài 92- Json Object Model – Customer – Service.....	97
Bài 93- Json Array Model – Group Customers (*)	97
Bài 94 – Cấu hình Routing	99
Bài 95 – Routing for Page Not Found	101
Module 9 – AngularJS (tt) - Form.....	102
Bài 96– Login Screen.....	102
Bài 97 – Form Đăng ký Khóa học	103
Bài 98 – Form Đăng ký Khóa học - Validation	103
Bài 99- Mathematics	104
Module 10 – AngularJS (tt) Triệu gọi External Restful API	105
Bài 100: Tương tác External Restful API- Ngân hàng Đông Á	105
Bài 101: Tương tác External Restful API- Product Service 1	112
Bài 102: Tương tác External Restful API- Product Service 2	117
Bài 103 – Coindesk API- Bitcoin Price Index in Real-time	120
Bài 104 – Get List of Public APIs.....	120
Bài 105 - Predict the gender of a person based on their name - API.....	121
Bài 106- Get US public data API	121

Bài 107- Random dog images API	121
Module 11 –Restful API với NodeJS và ExpressJS.....	122
Bài 108– Giải thích chi tiết mô hình hoạt động của MVC.....	122
Bài 109 – API và Restful API.....	122
Bài 110 – Tạo và thực thi Project Restful API.....	122
Bài 111 – Cài đặt nodemon cho Web Server Project.....	126
Bài 112 – Cài đặt morgan cho Web Server Project.....	128
Bài 113 – Tạo HTTP GET – List of Book.....	130
Bài 114 - Triệu gọi HTTP GET – List of Book.....	132
Bài 115 – Tạo HTTP GET – a Book	136
Bài 116– Triệu gọi HTTP GET – a Book	138
Bài 117 – Tạo HTTP POST – Create a Book.....	141
Bài 118 – Triệu gọi HTTP POST – Create a Book	144
Bài 119 – Tạo HTTP PUT – Update a Book.....	149
Bài 120 – Triệu gọi HTTP PUT – Update a Book	151
Bài 121– Tạo HTTP DELETE– Remove a Book.....	154
Bài 122 – Triệu gọi HTTP DELETE– Remove a Book	154
Bài 123 – Upload files to Server-1.....	157
Bài 124 – Upload files to Server-2.....	160
Bài 125 – Bài tập tổng hợp Restful API(*).....	165
Module 12 –Restful API với MongoDB	166
Bài 126- Cài đặt và sử dụng MongoDB	166
Bài 127 - Cấu hình và Kết nối NodeJS với MongoDB.....	174
Bài 128 – Tạo và triệu gọi API - HTTP GET – List Fashion	177
Bài 129 – Tạo và triệu gọi API - HTTP GET – A Fashion	181
Bài 130 – Tạo và triệu gọi API - HTTP POST – Create a Fashion	183
Bài 131 – Tạo và triệu gọi API - HTTP PUT – Update a Fashion.....	187
Bài 132 – Tạo và triệu gọi API - HTTP DELETE – Remove a Fashion	189
Bài 133 – Bài tập tổng hợp(*).....	190
Module 13 – Cookie và Session	192
Bài 134 – Trình bày ý nghĩa của Cookie và Session.....	192
Bài 135 – Lập trình Cookie	192
Bài 136 – Lập trình Cookie – Lưu thông tin đăng nhập	196
Bài 137 – Lập trình Session	197
Bài 138 – Lập trình Session – Bài tập tổng hợp Shopping Cart(*).....	199

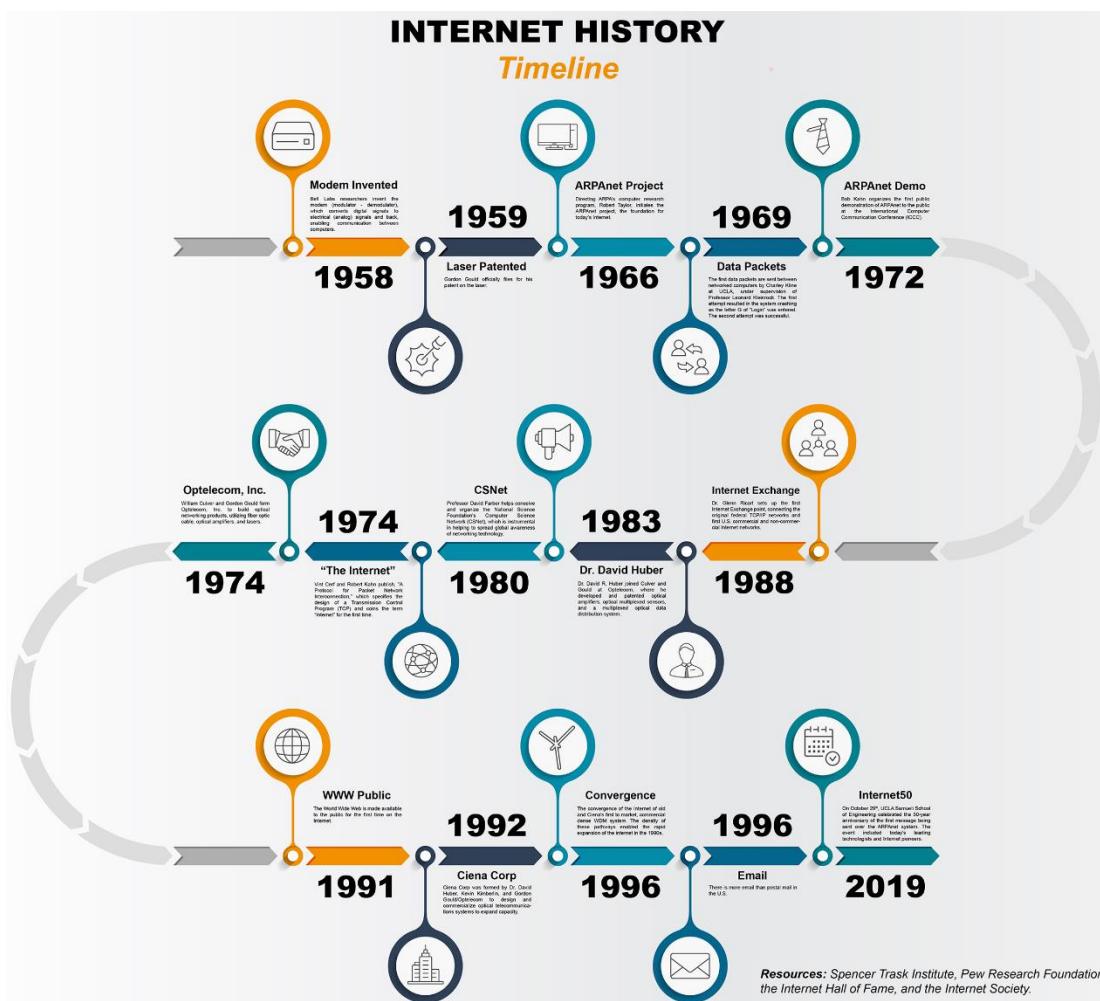
Module 1: Tổng quan về Phát triển Web kinh doanh

Nội dung kiến thức thực hành:

- + Ôn luyện các khái niệm cơ bản về Mạng và Internet
- + Khai thác tài nguyên Internet
- + Quy trình các bước để tạo lập một Website
- + Phân biệt được Website, Webpage, Server, WebServer
- + Phân biệt được Hosting-Domain, cách đăng ký
- + Nắm được quy trình triển khai và đưa Website vào hoạt động

Bài 1: History of the Internet

Hãy trình bày lịch sử phát triển của Internet



Bài 2:Server vs Client

Hãy phân biệt máy chủ(Máy Server) và máy khách (Máy Client). Máy để bàn và Laptop thường dùng hàng ngày có được gọi là Máy Server hay không? Vì sao?

Bài 3:HTTP vs HTTPS

Hãy trình bày và so sánh 2 giao thức HTTP và HTTPS?

Bài 4:Static Web and Dynamic Web

Hãy phân biệt Web Tĩnh và Web Động?

Bài 5:Hosting vs Domain

Hãy phân biệt Hosting và Domain? Có những loại Domain nào? Dựa vào đâu để chọn mua Hosting tối ưu cho hệ thống Website?. Nếu Website chuyên về thương mại điện tử thì nên mua những loại phổ biến nào:.org hay .edu hay .com hay .net hay .info hay .com.vn hay .vn ... (tự đề xuất thêm)

Bài 6:Server- Webserver-website

Hãy phân biệt Server, WebServer, Website, Webpage, Web Browser? Người ta nói một Server nên cài nhiều WebServer để tiết kiệm hệ thống đúng hay sai? Vì sao?

Bài 7: Structure of URL

Hãy hướng dẫn cách tìm địa chỉ IP của một domain?

Cho URL có cấu trúc: <https://tranduythanh.com/html-css-javascript/?id=113>

Hãy chỉ ra các thành phần: Protocol, domain, parameter trong link trên.

Bài 8:Quy trình tạo lập một Website

Hãy tìm hiểu nhiều kênh trên Internet để trình bày ít nhất 2 tư tưởng khác nhau về: Quy trình các bước để tạo lập một Website.

Module 2 – Hypertext Markup Language (HTML)

Nội dung kiến thức thực hành:

- + Sử dụng công cụ Visual Studio để tạo tập tin, project Web
- + Cấu hình IIS Webserver, cấu hình Hosting + Domain Free
- + Nắm được Cấu trúc của 1 tài liệu HTML
- + Thực hành Các thẻ (tags) cơ bản trong HTML
- + Thực hành Các thẻ danh sách trong HTML
- + Thực hành Thẻ liên kết trang
- + Cách tạo bảng trong HTML
- + Cách sử dụng Frame, Iframe, Frameset
- + Cách sử dụng Marquee
- + Cách sử dụng Multimedia

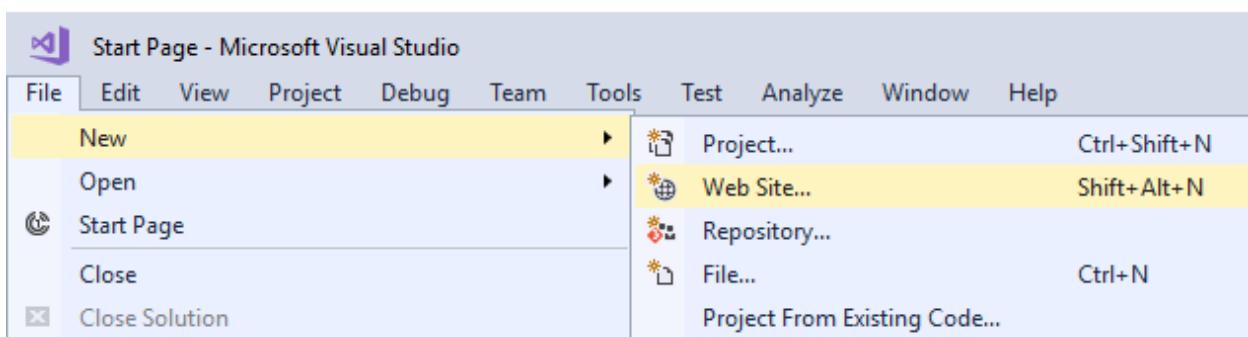
Bài 9: Sử dụng Visual Studio để tạo Website

Yêu cầu:

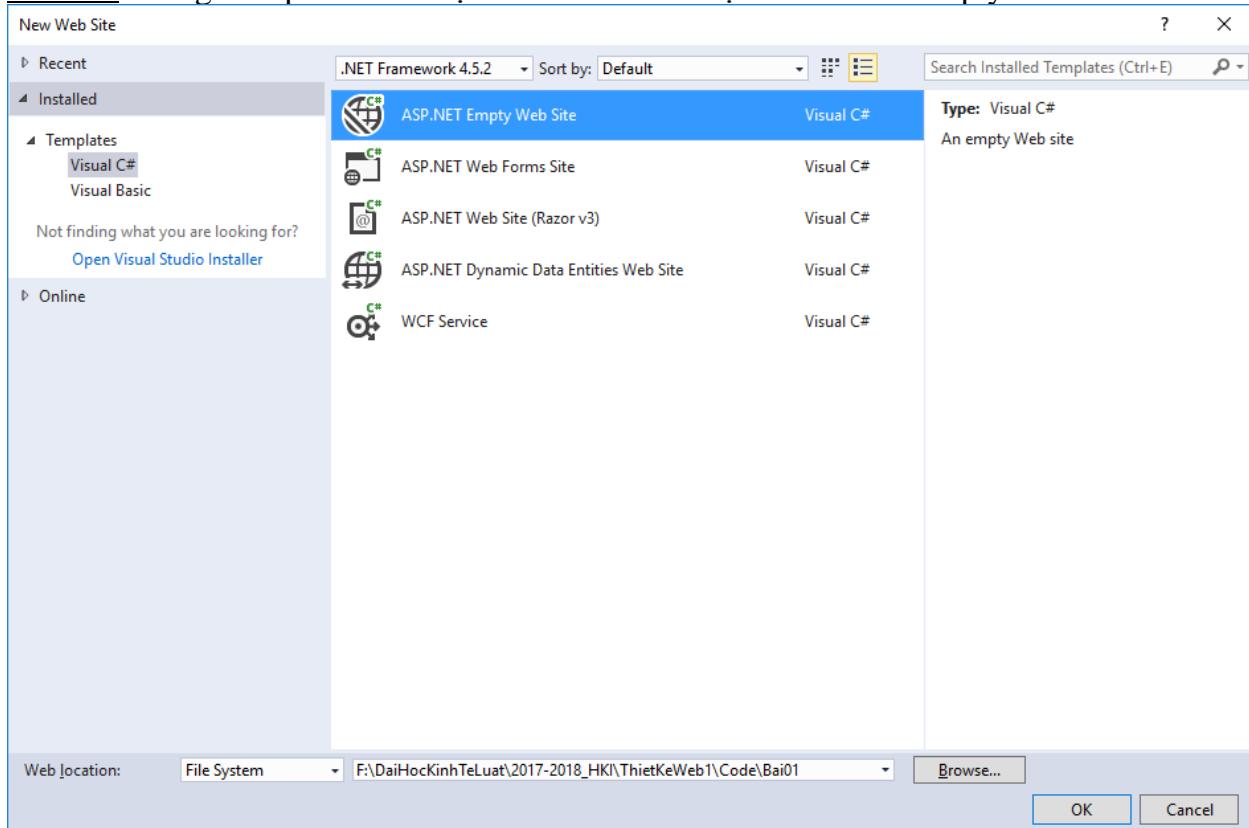
Sử dụng Công cụ Visual Studio để tạo một Project Web, cách thức tạo và chạy một File HTML trong Visual Studio.

Hướng dẫn:

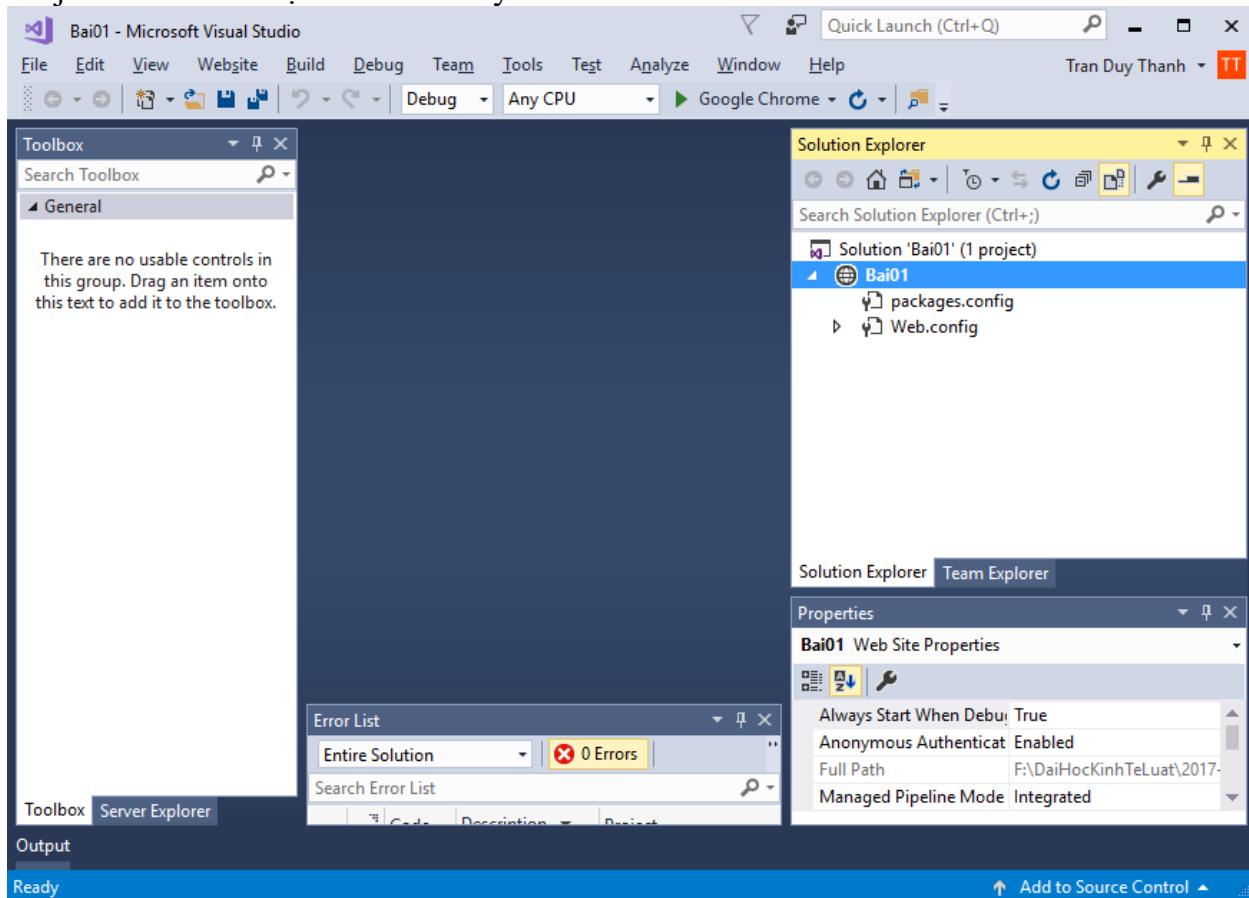
- + Bước 1: Khởi động Microsoft Visual Studio
- + Bước 2: Vào File/ chọn New / Chọn Web Site... (hoặc nhấn tổ hợp phím Shift + Alt+ N) để tạo một Project Website:



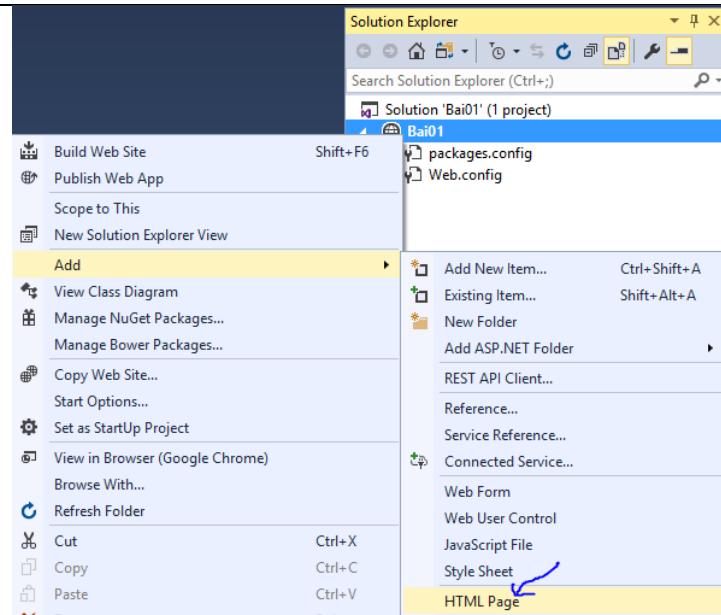
Bước 3: Trong Templates → Chọn Visual C# → Chọn ASP.NET Empty Web Site



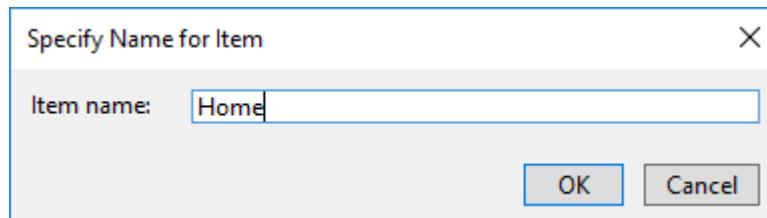
Mục Web Location: Chọn Nơi lưu trữ, đặt tên Bai01 rồi nhấn nút OK, màn hình Project Web xuất hiện như dưới đây:



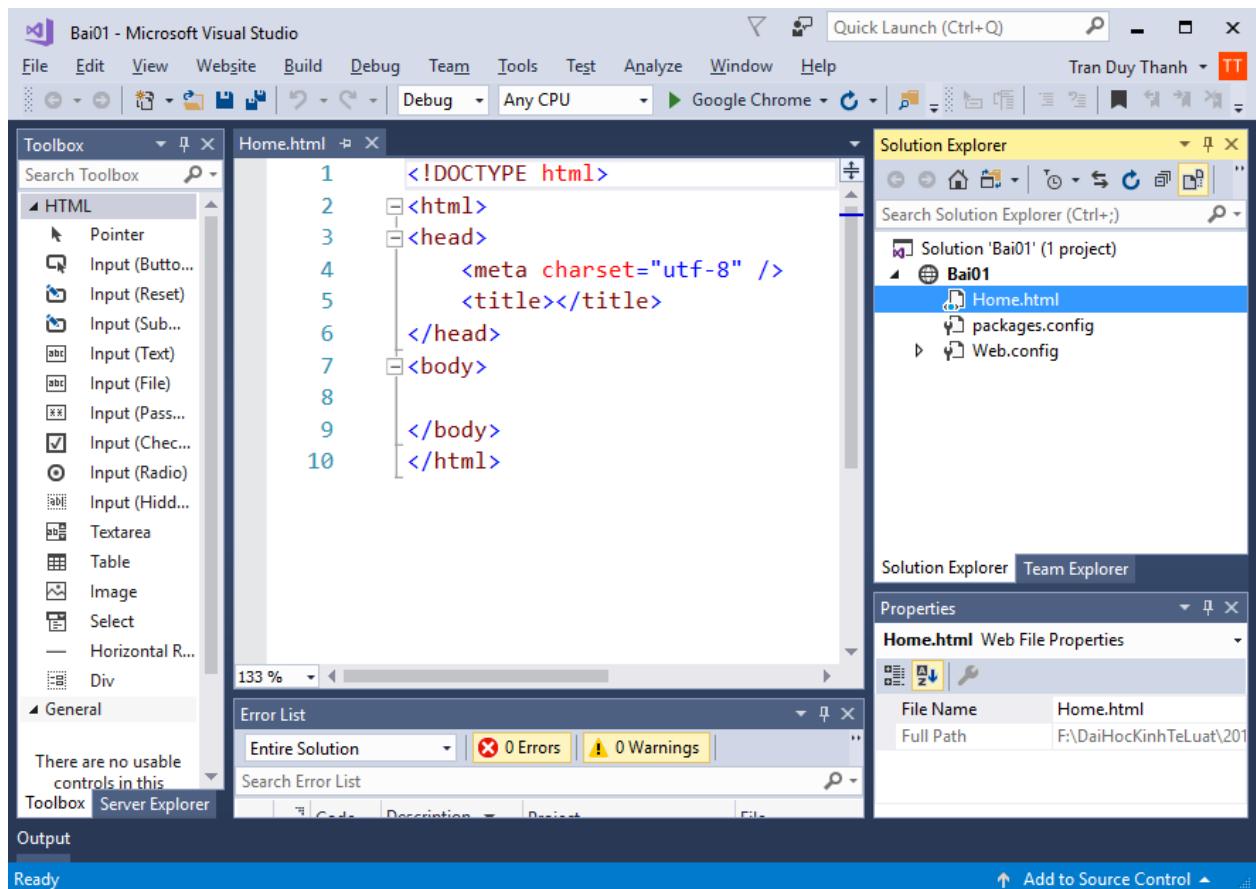
Bước 4: Trong cửa sổ Solution Explorer → bấm chuột phải chọn Add/Chọn HTML Page:



Sau khi chọn HTML Page, Visual Studio sẽ hiển thị cửa sổ đặt tên cho trang Web tĩnh HTML (lưu ý: Nếu không thấy HTML page ở đây thì chọn Add New Item → chọn HTML):



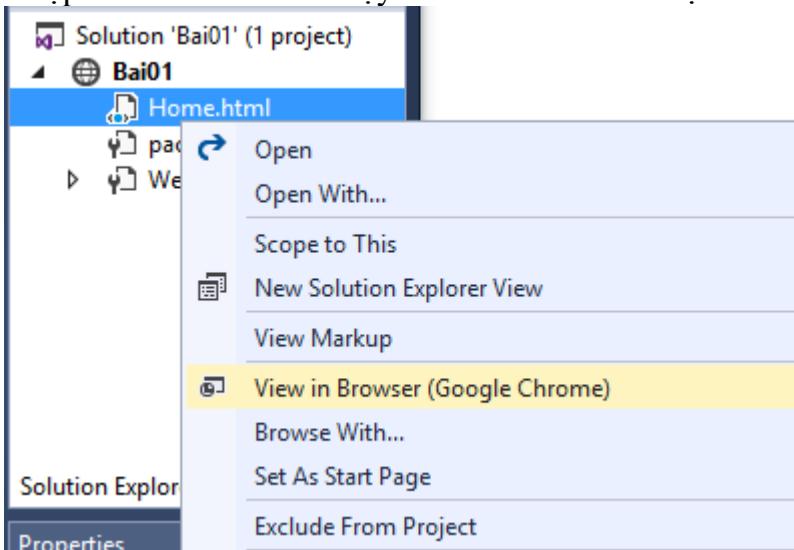
Ta đặt Home rồi bấm nút OK, Trang Web tĩnh Home.html xuất hiện như bên dưới:



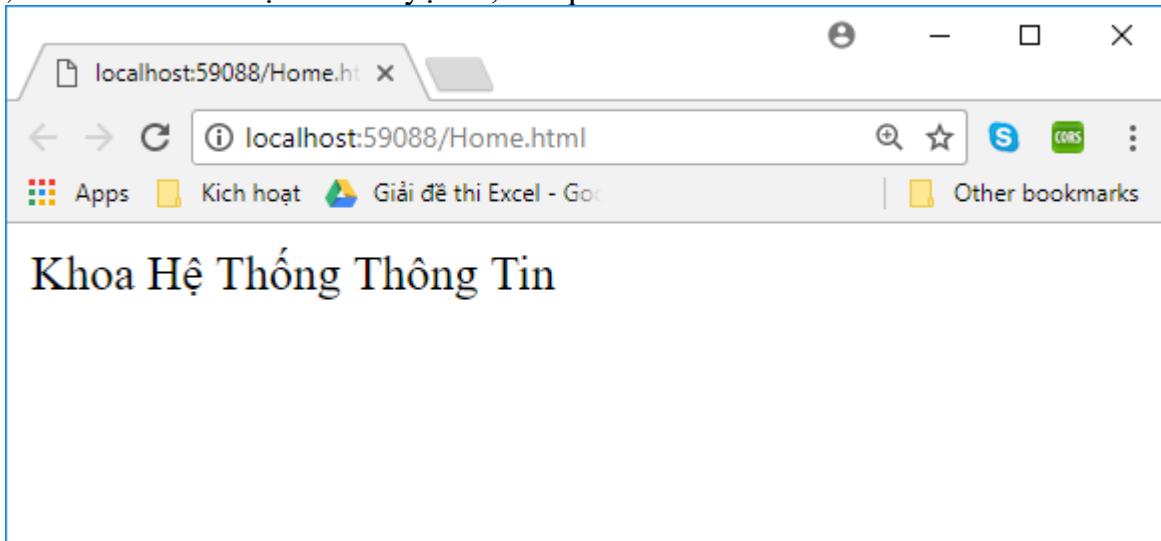
Trong thẻ Body nhập chuỗi “Khoa Hệ Thống Thông Tin”:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title></title>
</head>
<body>
    Khoa Hệ Thống Thông Tin
</body>
</html>
```

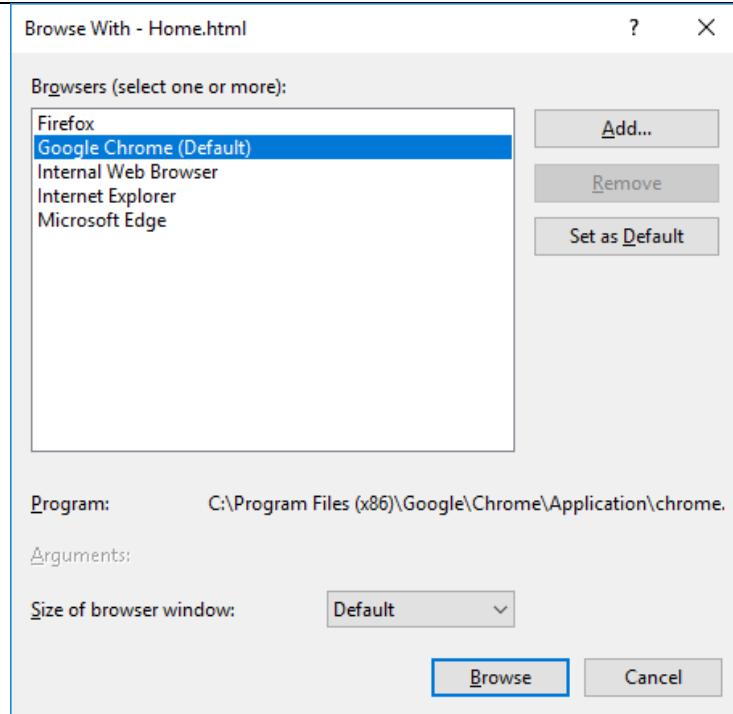
Bước 5: Để chạy Website trong Visual studio ta có nhiều cách, cách đơn giản nhất là bấm chuột phải vào tập tin HTML muốn chạy View In Browser hoặc Browser With:



Nếu chọn View In Browser (tùy vào trình duyệt bạn đang chạy mặc định trong máy tính), Visual sẽ kích hoạt trình duyệt đó, kết quả:



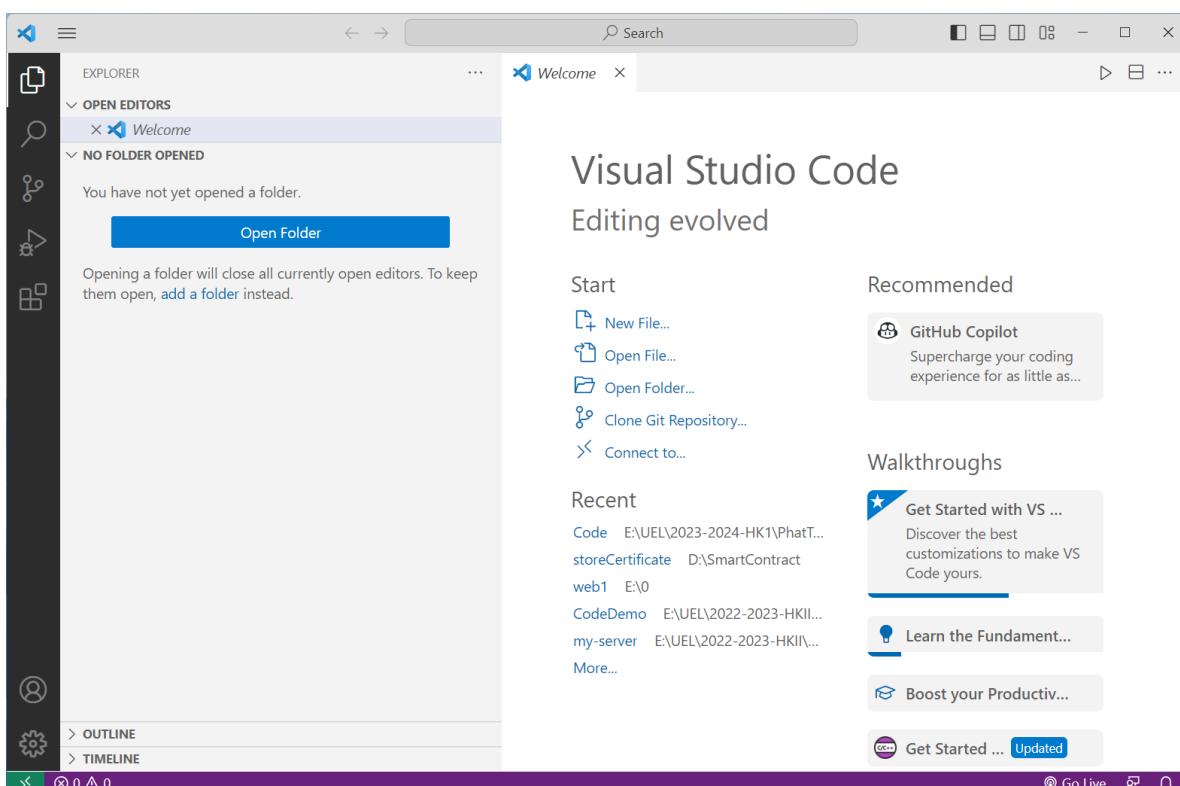
Nếu chọn Browse With: Visual Studio sẽ hiển thị cửa sổ yêu cầu ta lựa chọn Trình duyệt khác:



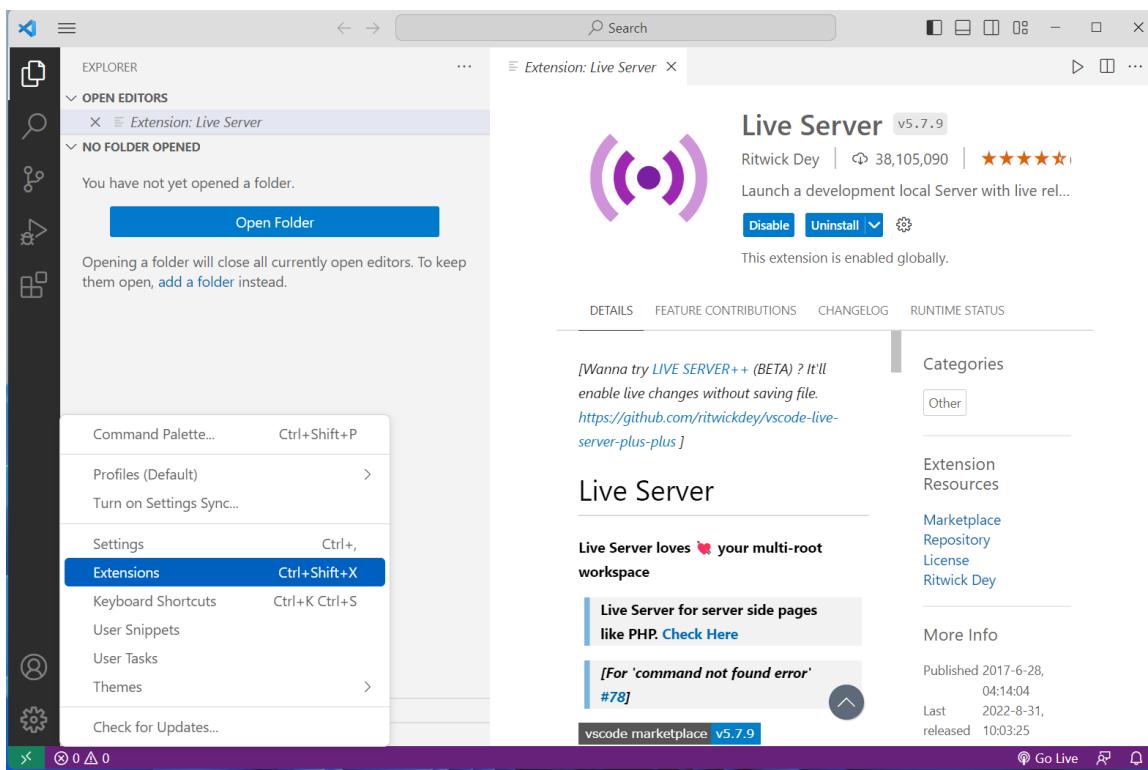
Ví dụ màn hình ở trên, máy tính có 5 Trình duyệt, ta muốn Trình duyệt nào chạy mặc định chỉ chọn nó vào bấm “Set as Default”, muốn chạy Website này thì bấm vào nút Browse ở bên dưới cửa sổ.

Dưới đây là hướng dẫn tạo HTML project trong Visual Studio Code (và xuyên suốt các bài tập của môn học này ta sử dụng Visual Studio Code, kể cả bài thi giữa kỳ và đồ án kết thúc môn học).

Step 1: Tải và cài đặt Visual Studio Code:
<https://code.visualstudio.com/>

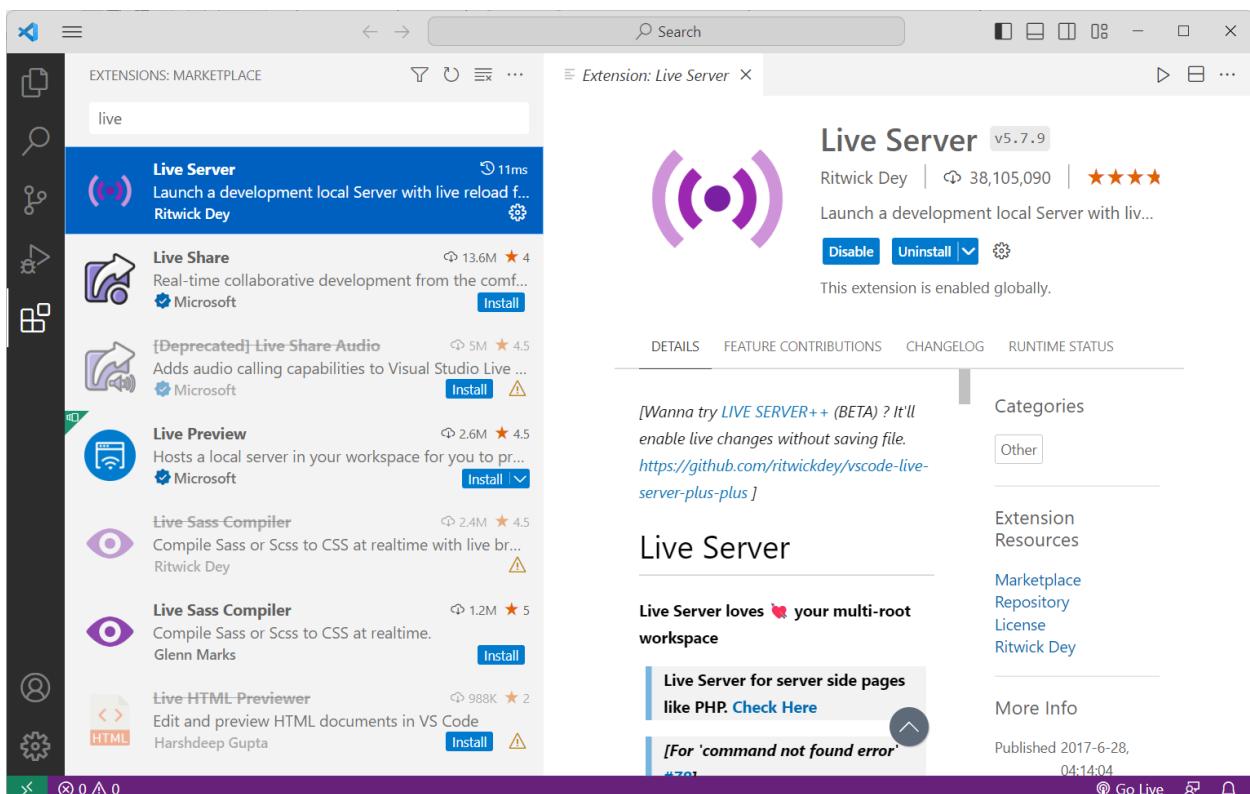


Step 2: Cài đặt Live Server extension



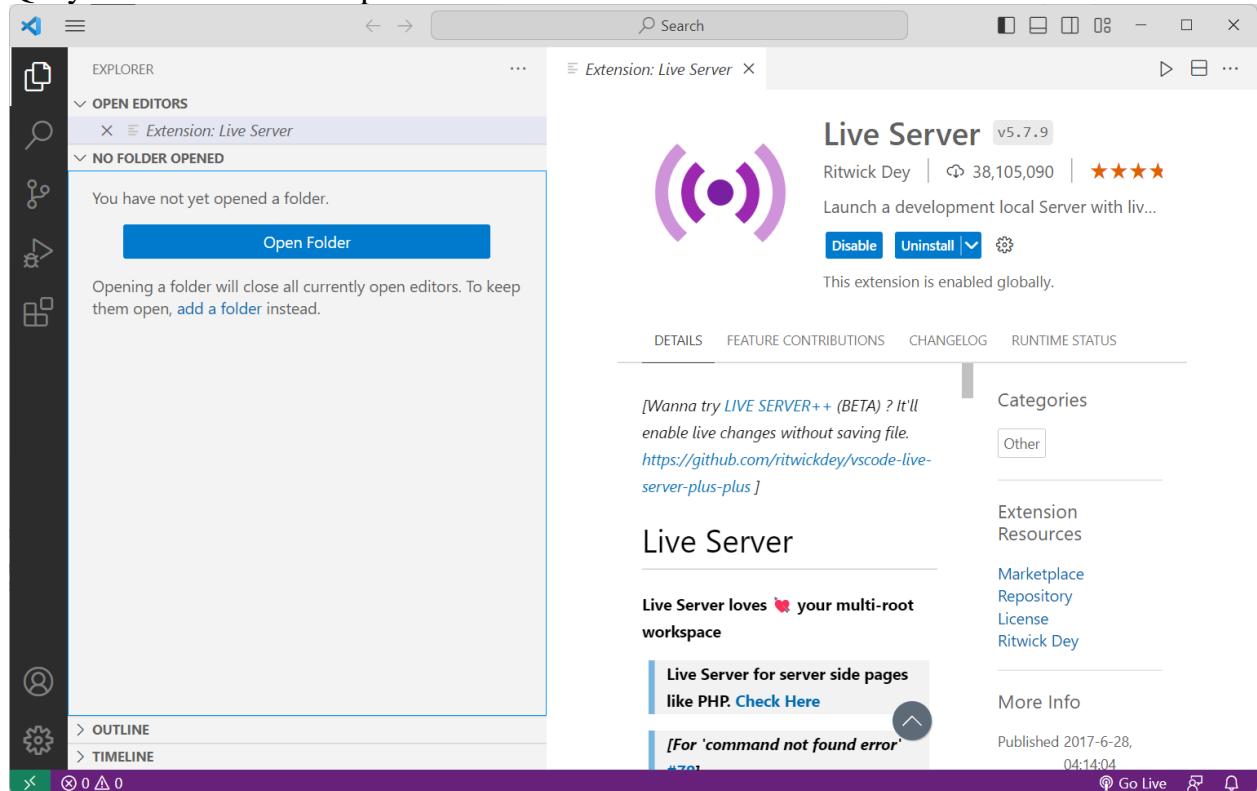
Nhấn chọn nút biểu tượng “Manage” ở góc trái bên dưới màn hình, sau đó chọn “Extensions”.

Tìm từ khóa “**Live Server**” và bấm **install**:

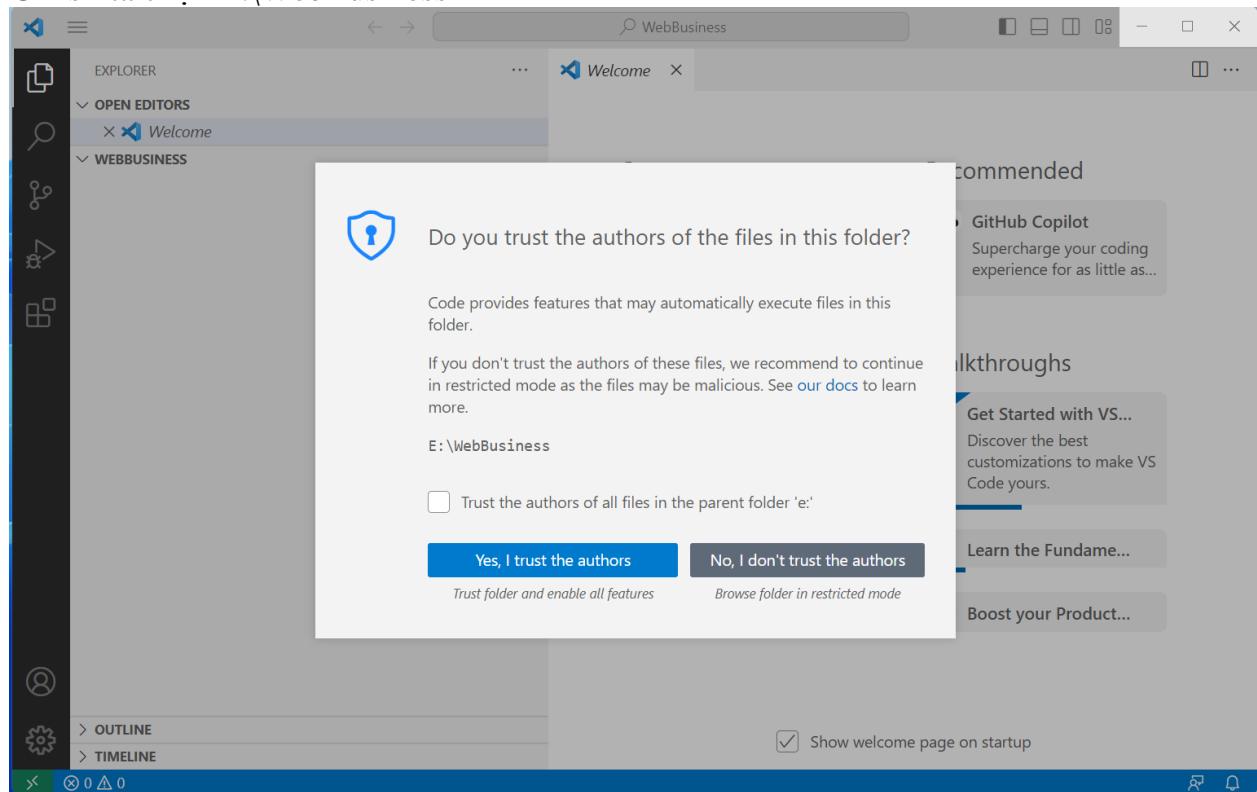


Step 3: Tạo Folder/Project cho HTML

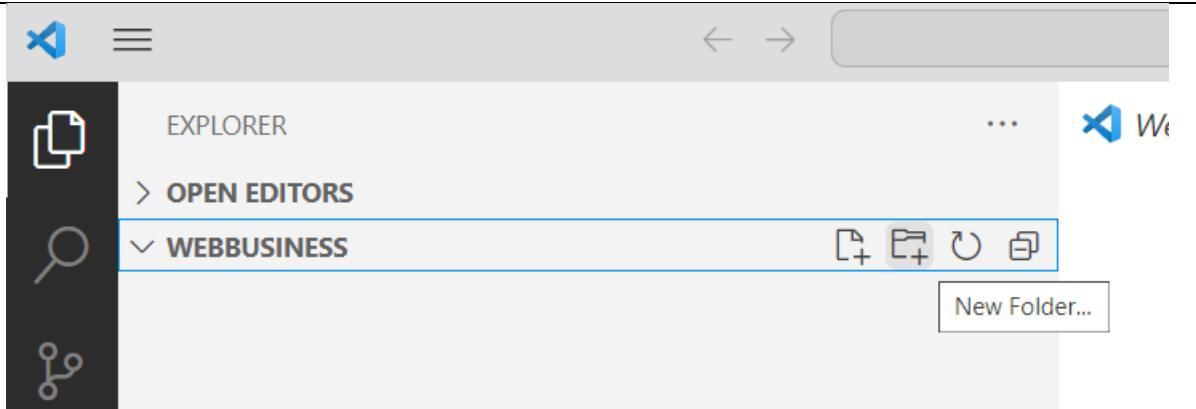
Quay trở về màn hình Explorer:



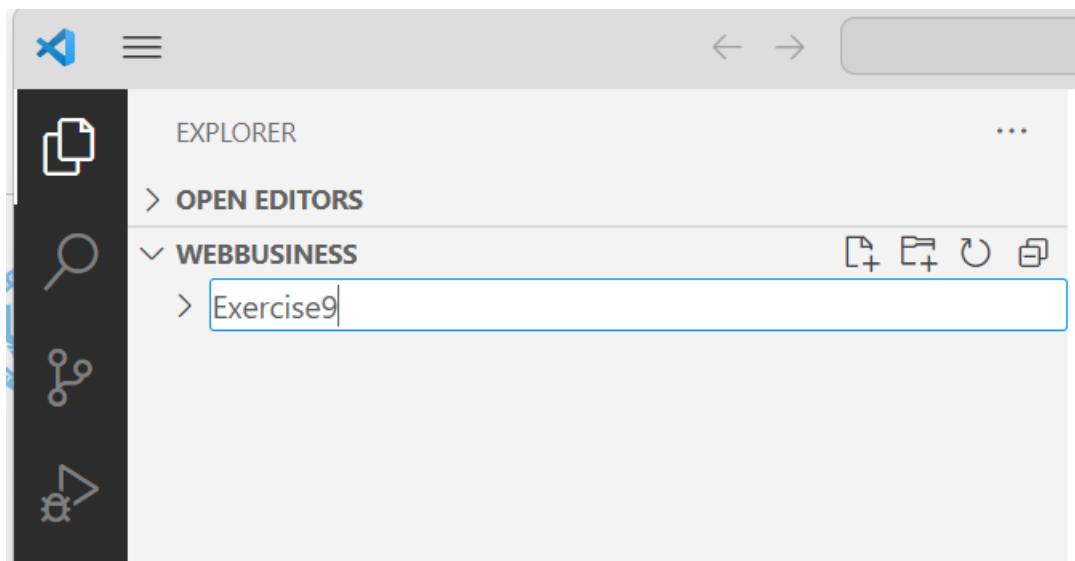
Nhấn nút “Open Folder” để thiết lập thư mục soạn thảo lập trình.
Giả sử ta chọn E:\WebBusiness



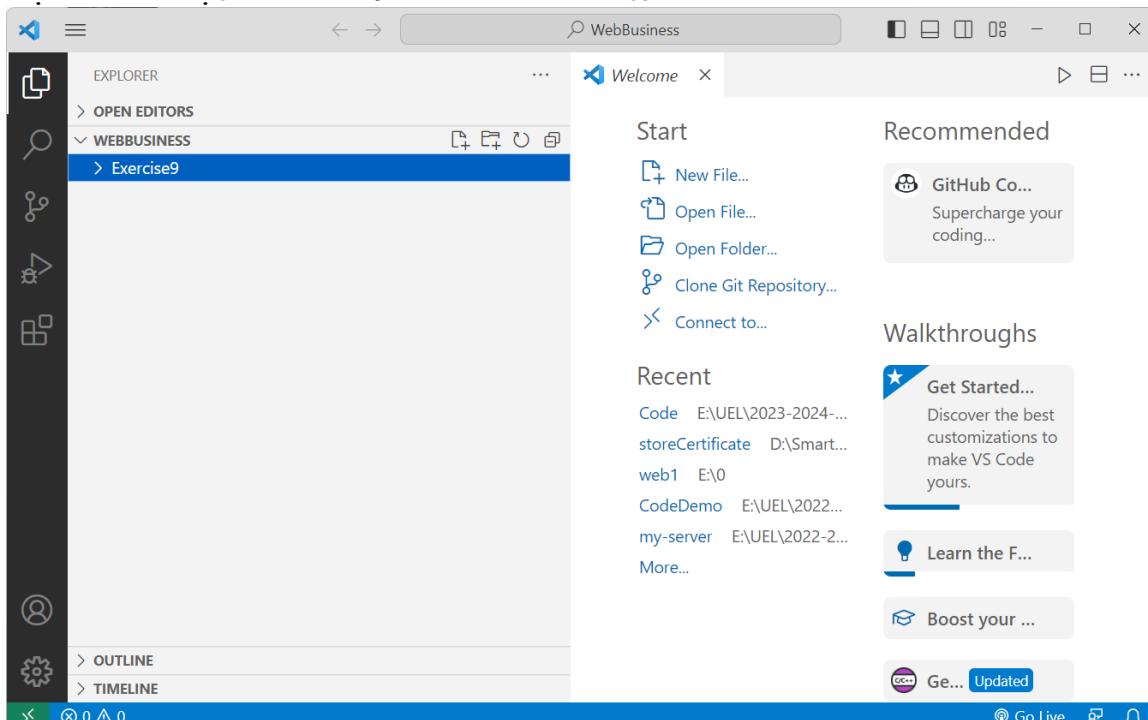
Ta nhấn “Yes I trust the authors” nếu mà nhìn Trust the Authors hiển thị như trên.



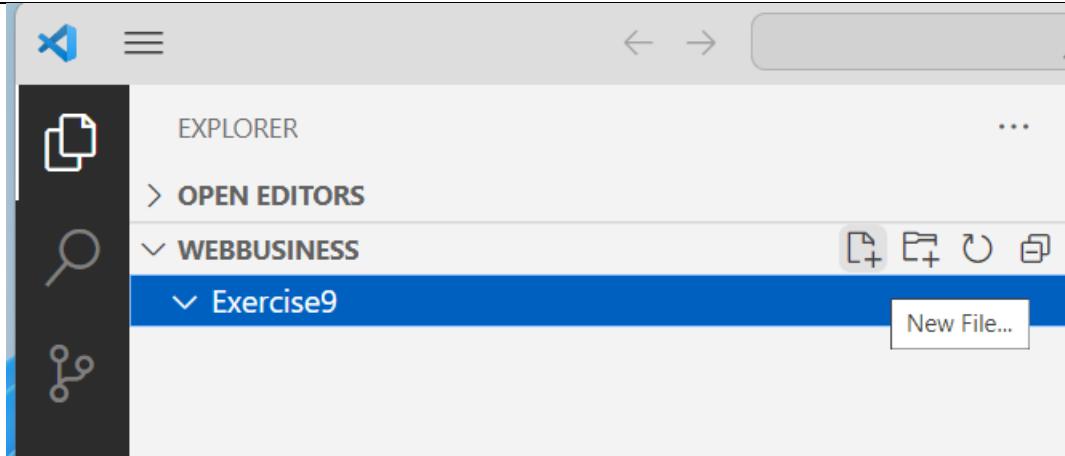
Nhấn chọn “New Folder” button như hình ở bên trên.



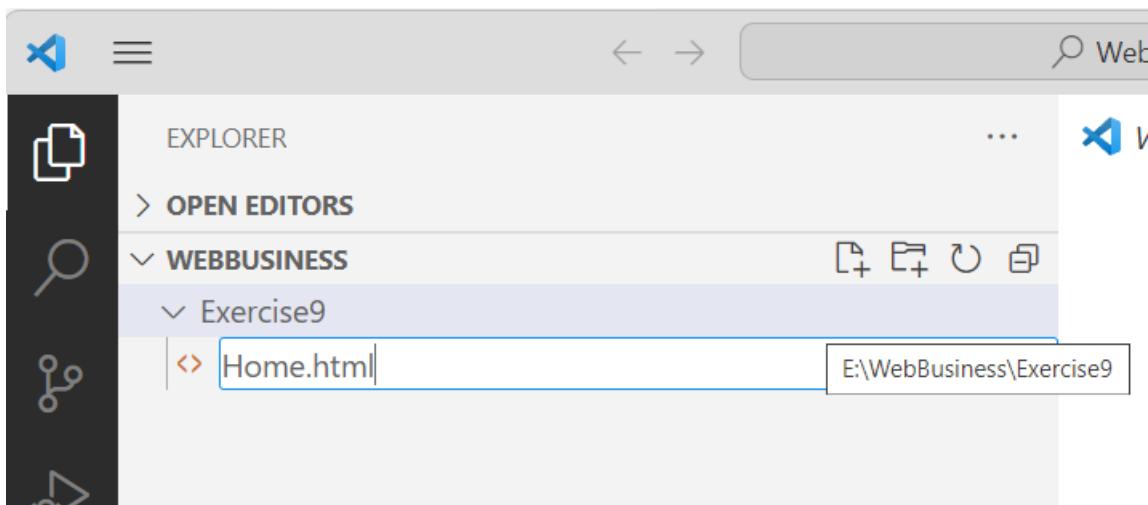
Đặt tên thư mục “Exercise9” sau đó nhấn Enter



Tiếp tục tạo file HTML trong **Exercise9**, nhấn nút create New File:



Nhấn nút “New File...” như hình trên



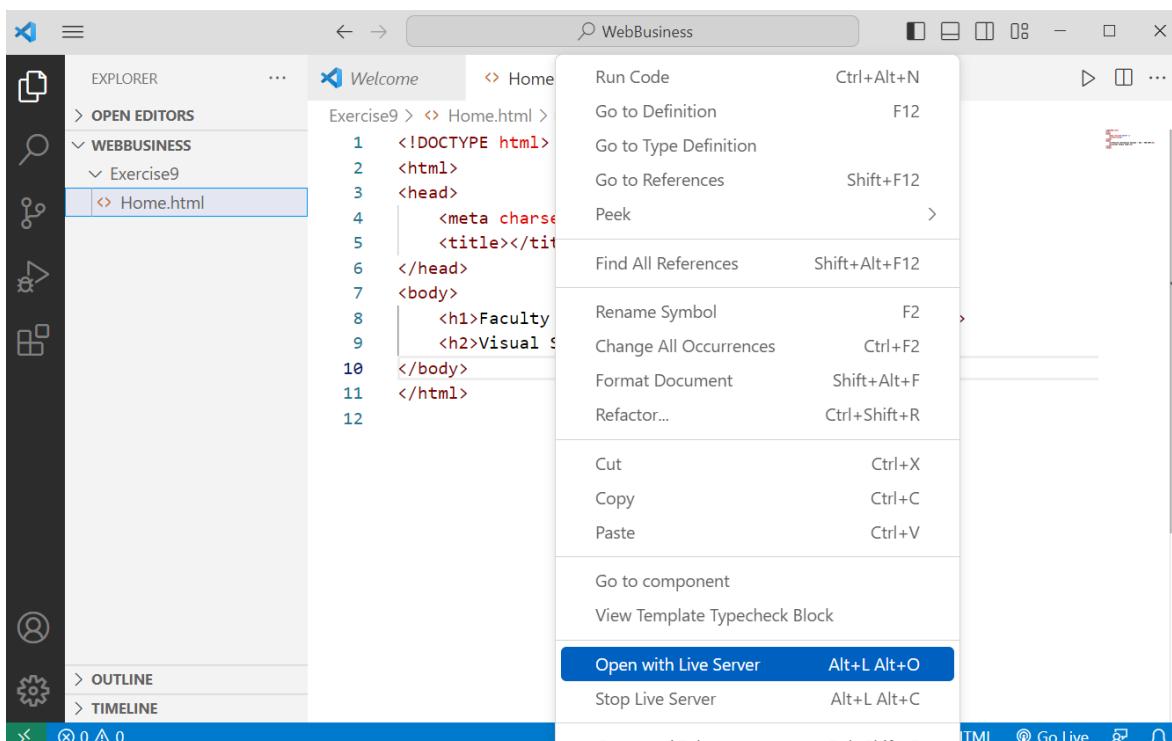
Nhập tên file “Home.html” rồi nhấn Enter

Home.html được tạo ra như dưới đây, sau đó ta bổ sung một số HTML code như dưới đây:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title></title>
</head>
<body>
    <h1>Faculty Information Systems - UEL - VNU HCM</h1>
    <h2>Visual Studio Code</h2>
</body>
</html>
```

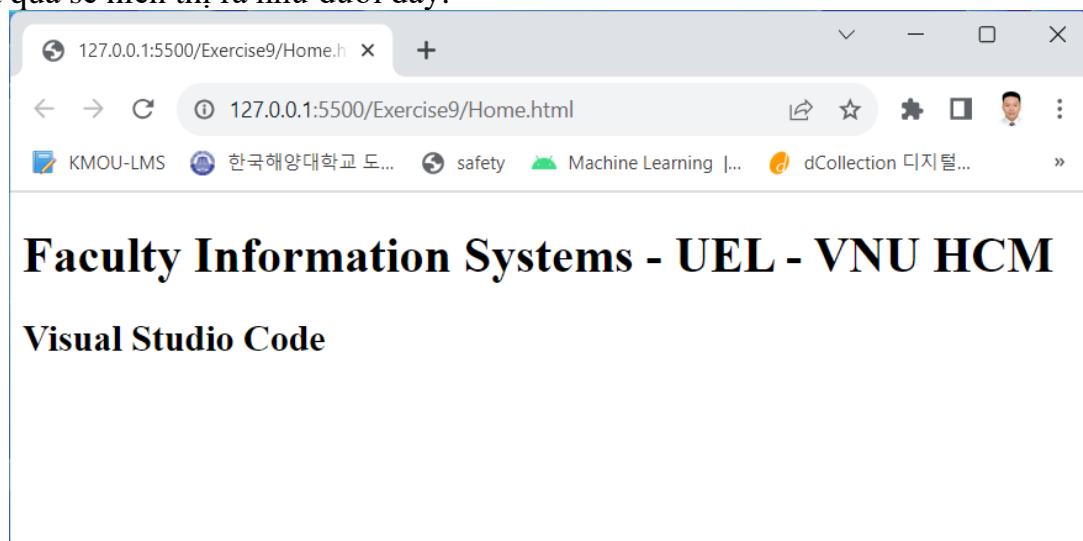
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title></title>
</head>
<body>
    <h1>Faculty Information Systems - UEL - VNU HCM</h1>
    <h2>Visual Studio Code</h2>
</body>
</html>
```

Step 4: Chạy HTML project



Bấm chuột phải vào “**Home.html**” rồi click chọn “**Open with Live Server**”

Kết quả sẽ hiển thị ra như dưới đây:



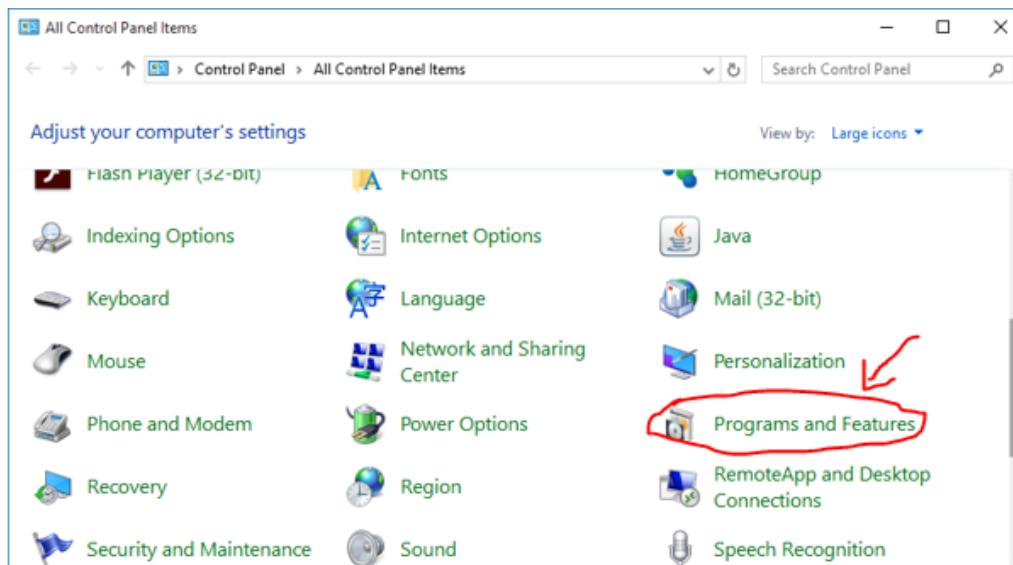
Bài 10: Cấu hình IIS Webserver

Yêu cầu:

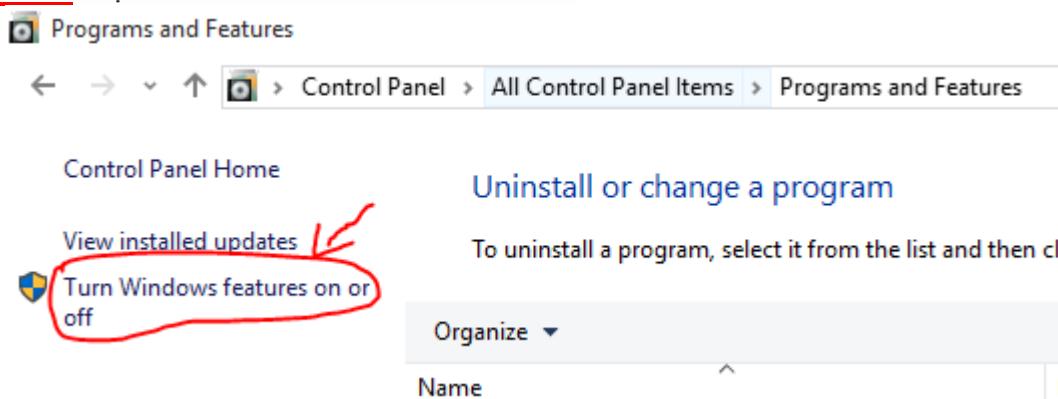
Hãy cấu hình IIS Webserver và deploy bài tập 1 vào WebServer để Website này có thể được chạy trên mạng LAN.

Hướng dẫn:

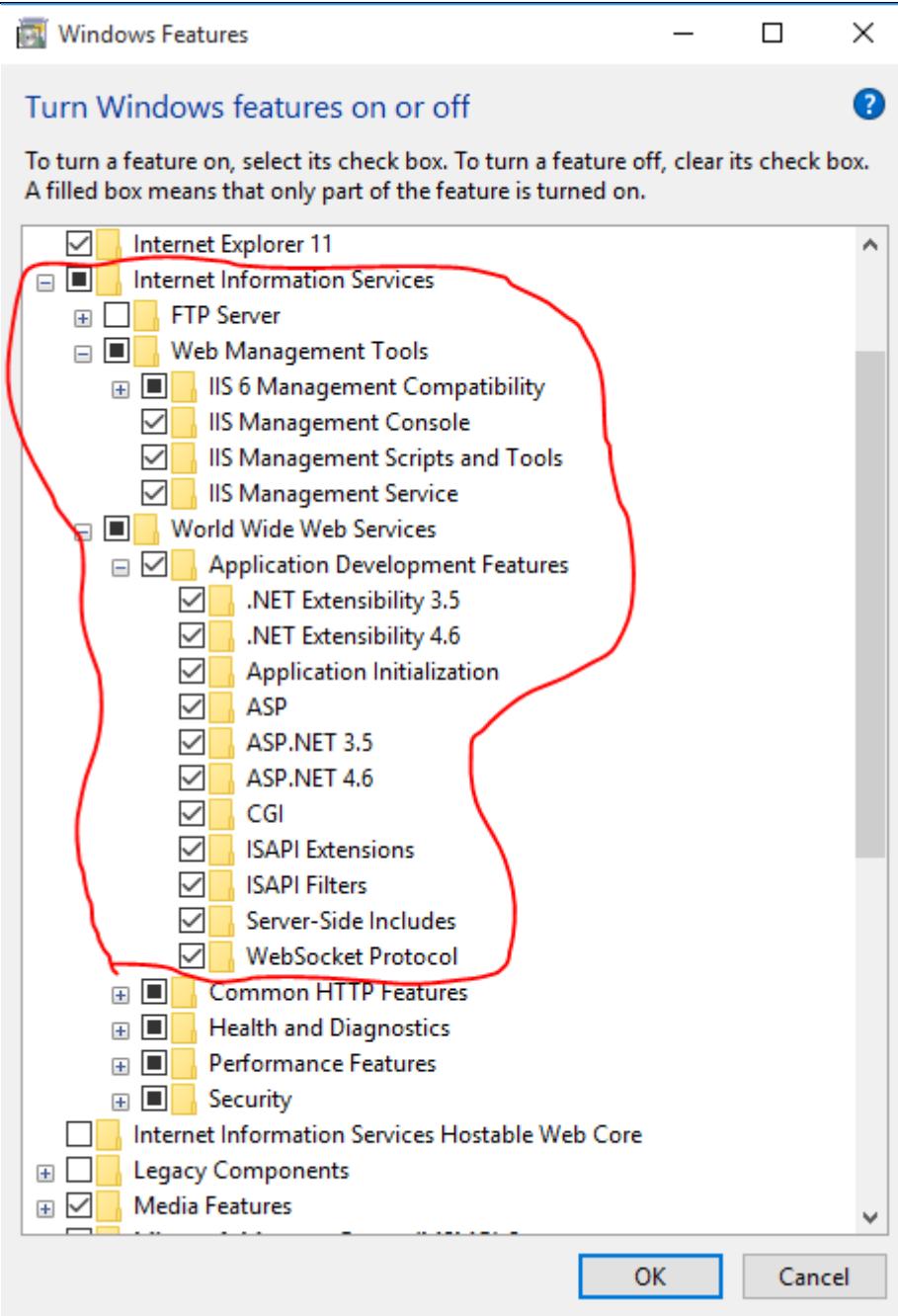
Bước 1: Vào Control panel chọn Programs and Features



Bước 2: Chọn Turn Windows features on or off:

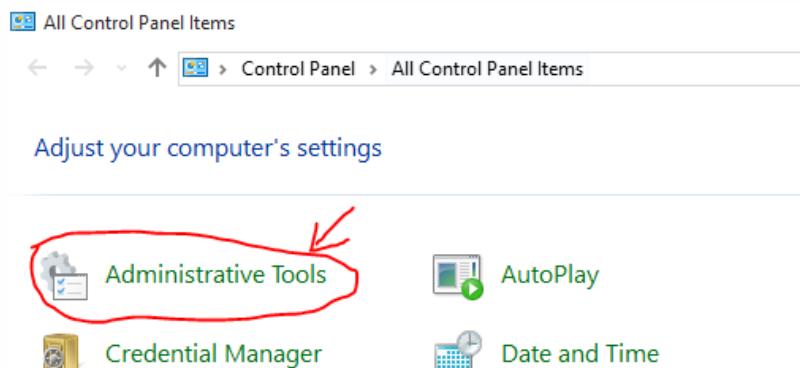


Bước 3: Tiến hành lựa chọn để cài IIS Web Server theo màn hình dưới đây:



Sau khi lựa chọn giống như khung khoanh màu đỏ, nhấn OK để tiến hành cài đặt, tùy thuộc vào máy mà chương trình có thể cài từ 5-10 phút.

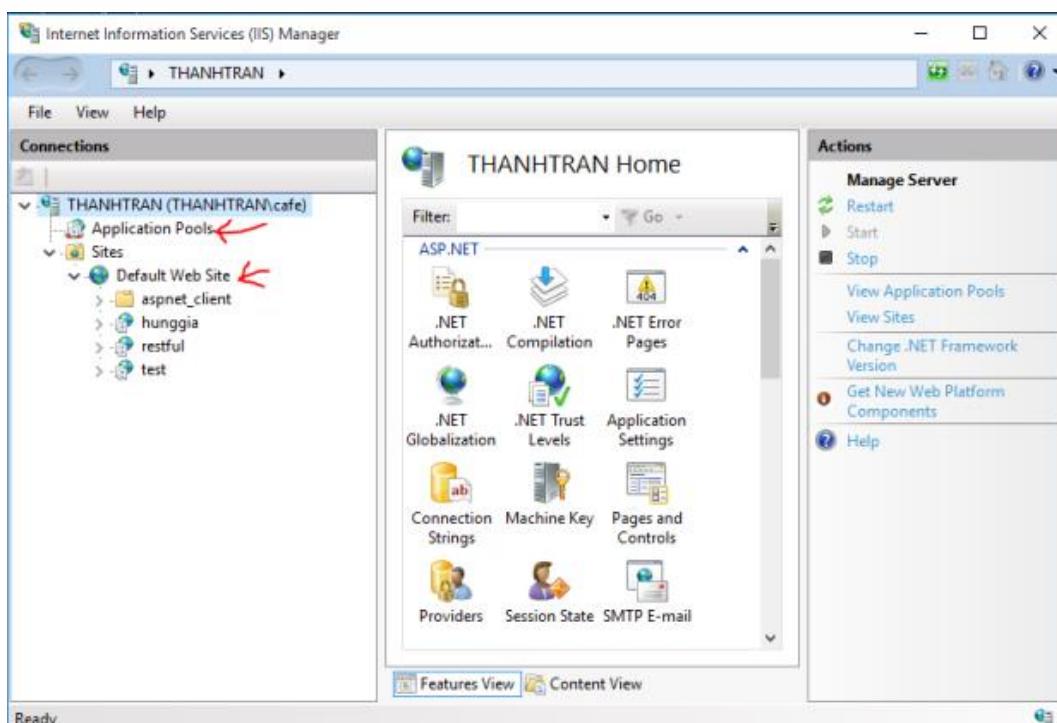
Bước 4: Sử dụng IIS Web Server: Sau khi cài xong IIS Web Server, bạn quay lại màn hình Control panel, bạn sẽ thấy Administrative Tools:



Bạn click vào biểu tượng Administrative Tools, ta có giao diện tiếp theo:

Control Panel > All Control Panel Items > Administrative Tools				
Name	Date modified	Type	Size	
Component Services	7/10/2015 5:59 PM	Shortcut	2 KB	
Computer Management	7/10/2015 5:59 PM	Shortcut	2 KB	
Defragment and Optimize Drives	7/10/2015 5:59 PM	Shortcut	2 KB	
Disk Cleanup	7/10/2015 6:01 PM	Shortcut	2 KB	
Event Viewer	7/10/2015 5:59 PM	Shortcut	2 KB	
Internet Information Services (IIS) Manager	7/10/2015 6:00 PM	Shortcut	2 KB	
iSCSI Initiator	7/10/2015 6:00 PM	Shortcut	2 KB	

Bạn chọn “Internet Information Services (IIS) Manager” như hình trên:



Trong màn hình quản trị bạn thấy có 2 vùng: Application Pools và Default Web site.

Bước 4.1 Cấu hình Application Pools:

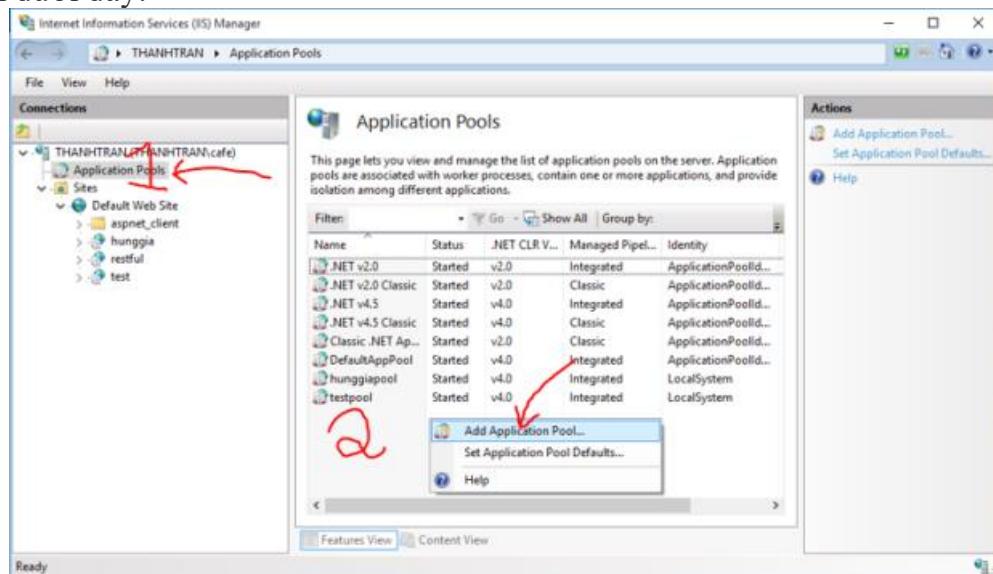
Application Pool là gì?

Application Pool có thể chứa một hoặc nhiều ứng dụng và cho phép chúng ta cấu hình cấp độ giữa các ứng dụng web khác nhau. Ví dụ, nếu bạn muốn cài đặt tất cả các ứng dụng web chạy trong cùng một máy, bạn có thể làm điều này bằng cách tạo ra Application Pool riêng biệt cho mỗi ứng dụng web và đặt chúng trong Application Pool tương ứng. Bởi vì mỗi Application Pool chạy trong quá trình làm việc riêng của mình, các lỗi trong Application Pool sẽ không ảnh hưởng đến các ứng dụng đang chạy trong Application Pool khác. Triển khai ứng dụng trong Application Pool là lợi thế chính của

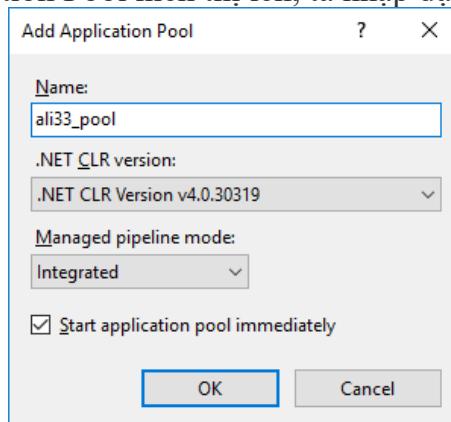
IIS trong quá trình làm việc ở chế độ cách ly bởi vì bạn có thể tùy chỉnh Application Pool để đạt được cấp độ tách biệt ứng dụng mà bạn cần.

Khi bạn cấu hình Application Pool để sẵn sàng tối ưu, bạn cũng nên xem xét làm thế nào để cấu hình Application Pool bảo mật ứng dụng. Ví dụ, bạn có thể cần phải tạo ra Application Pool riêng cho ứng dụng đòi hỏi mức độ bảo mật cao, trong khi cho phép các ứng dụng đòi hỏi một mức độ thấp hơn của bảo mật để chia sẻ cùng Application Pool.

Bạn bấm chuột phải vào màn hình Application Pools/ chọn Add Application Pool... như hình dưới đây:



Màn hình tạo mới Application Pool hiển thị lên, ta nhập đại “ali33_Pool” rồi bấm OK:



Sau khi bấm OK, bạn quan sát:

Application Pools

This page lets you view and manage the list of application pools on the server. Application pools are associated with worker processes, contain one or more applications, and provide isolation among different applications.

Name	Status	.NET CLR V...	Managed Pipel...	Identity	Applications
.NET v2.0	Started	v2.0	Integrated	ApplicationPoolId...	0
.NET v2.0 Classic	Started	v2.0	Classic	ApplicationPoolId...	0
.NET v4.5	Started	v4.0	Integrated	ApplicationPoolId...	0
.NET v4.5 Classic	Started	v4.0	Classic	ApplicationPoolId...	0
ali33_pool	Started	v4.0	Integrated	ApplicationPoolId...	0
Classic .NET AppPool	Started	v2.0	Classic	ApplicationPoolId...	0
DefaultAppPool	Started	v4.0	Integrated	ApplicationPoolId...	3
foodmanagerpool	Started	v4.0	Integrated	LocalSystem	1
FoodServer_Pool	Started	v4.0	Integrated	LocalSystem	0
hunggiapool	Started	v4.0	Integrated	LocalSystem	5
restpool	Started	v4.0	Integrated	LocalSystem	1
testpool	Started	v4.0	Integrated	LocalSystem	3
tui_pool	Started	v4.0	Integrated	LocalSystem	1

Actions

- Add Application Pool
- Set Application Pool
- Application Pool Tasks
- Start
- Stop
- Recycle...
- Edit Application Pool
- Basic Settings...
- Recycling...
- Advanced Settings...**
- Rename
- Remove
- View Applications
- Help

Ở màn hình trên bạn tiếp tục chọn Advanced Settings.. cho FoodServer_Pool:

Advanced Settings

Start Mode	OnDemand
CPU	
Limit (percent)	0
Limit Action	NoAction
Limit Interval (minutes)	5
Processor Affinity Enabled	False
Processor Affinity Mask	4294967295
Processor Affinity Mask (64-bit c	4294967295
Process Model	
> Generate Process Model Event L	
Identity	ApplicationPoolIdentity

Application Pool Identity

Built-in account: LocalSystem

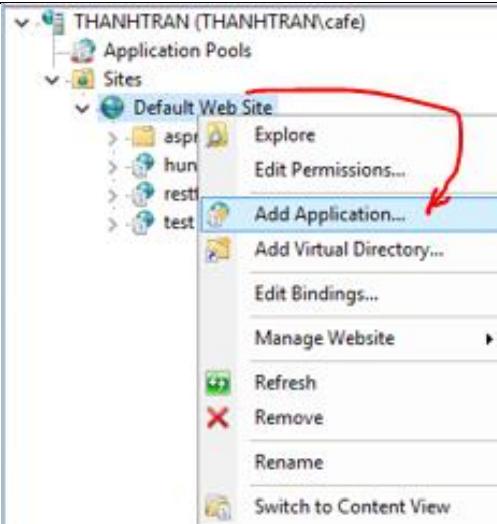
Custom account: Set...

OK Cancel

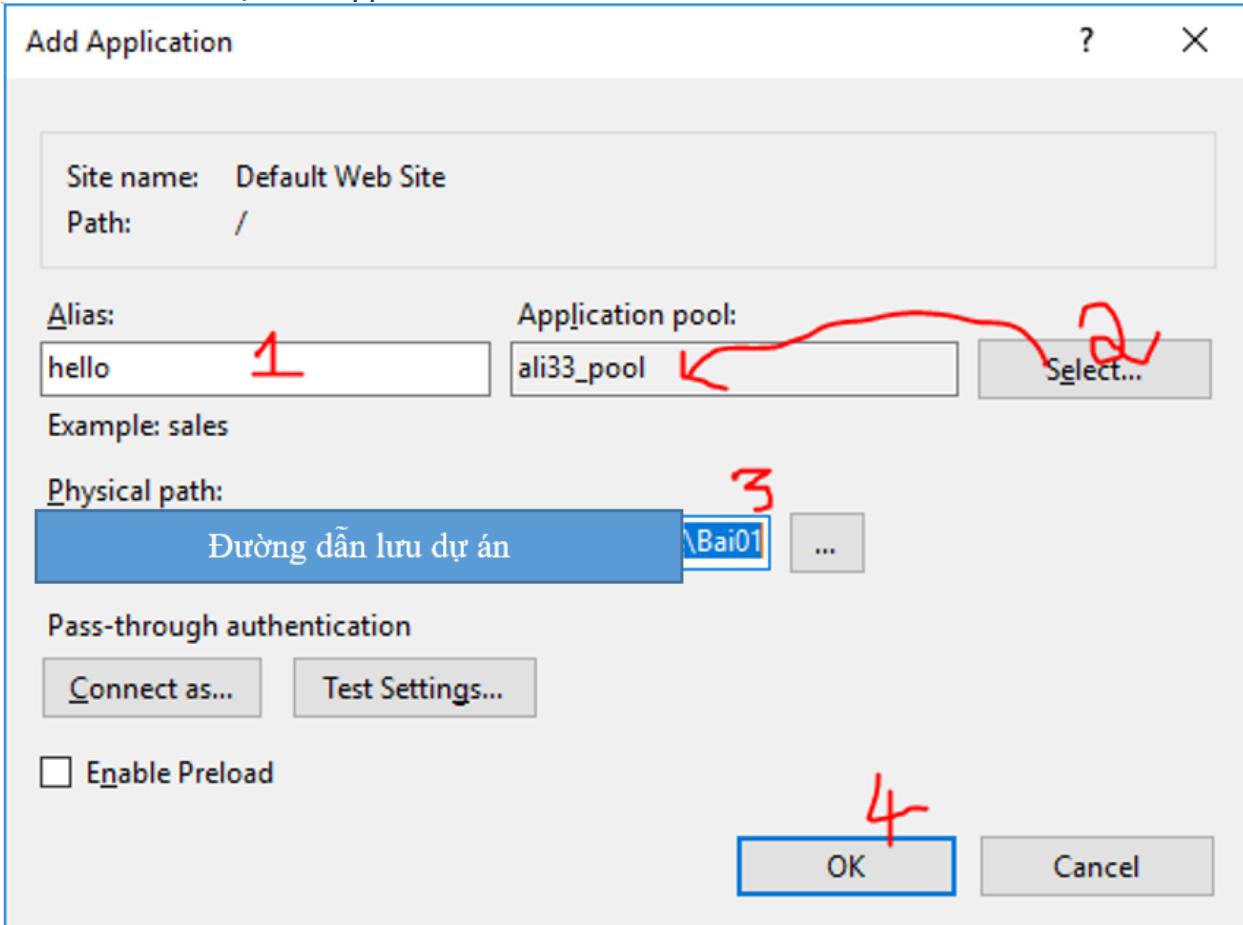
OK Cancel

Bạn cần chỉnh ApplicationPoolIdentity qua **LocalSystem**.

Bước 4.2: Cấu hình WebService lên IIS Server
Bấm chuột phải vào Default Web Site / chọn Add Application ...



Màn hình hiển thị Add Application:



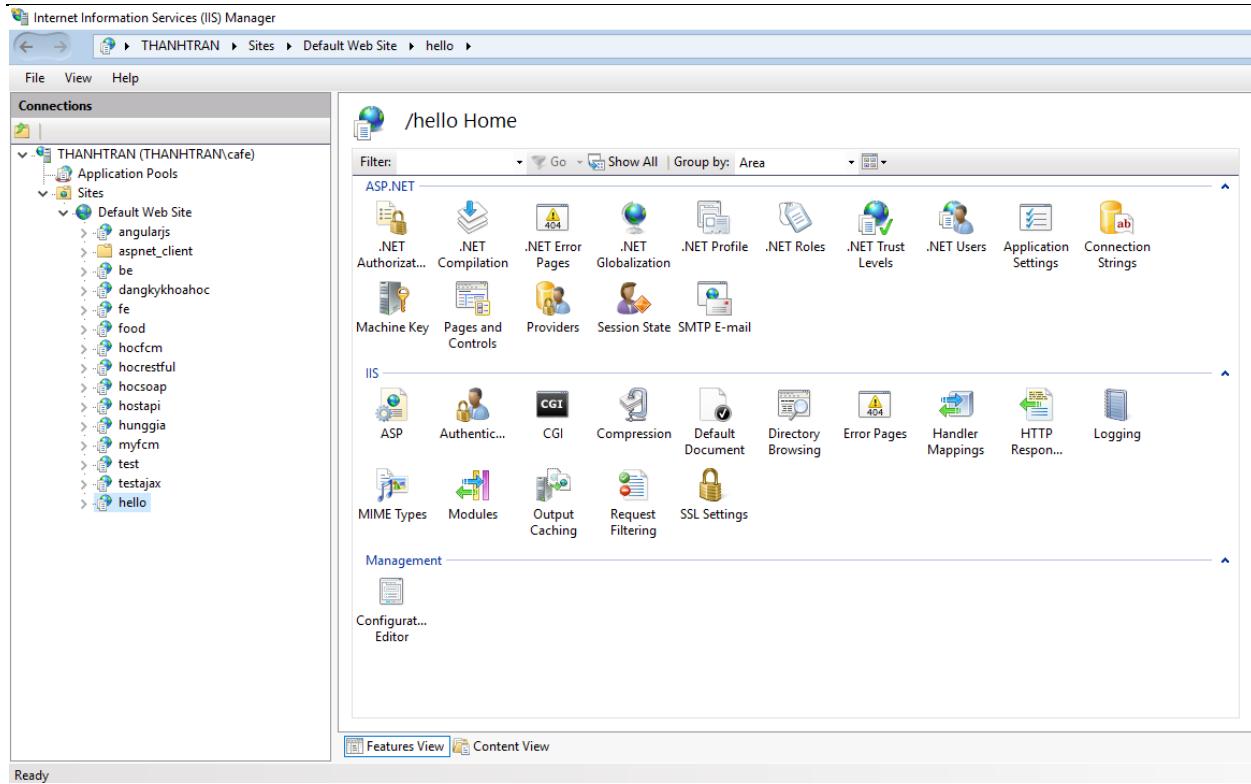
Mục Alias (1): Đặt tên cho Website, ở đây ta đặt **hello**

Mục Application pool (2): Nhấn nút Select... và chọn đúng **ali33_pool**

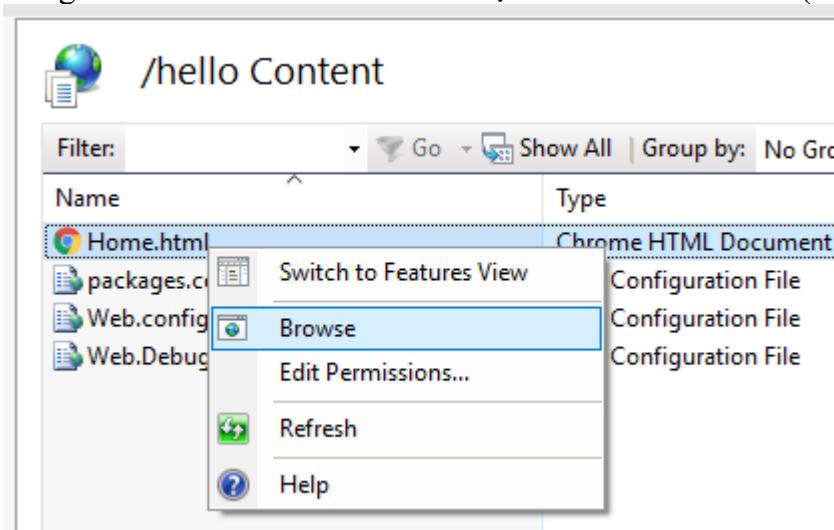
Mục Physical path (3): Trỏ tới đường dẫn chứa source code của project

Sau đó bấm OK để tạo.

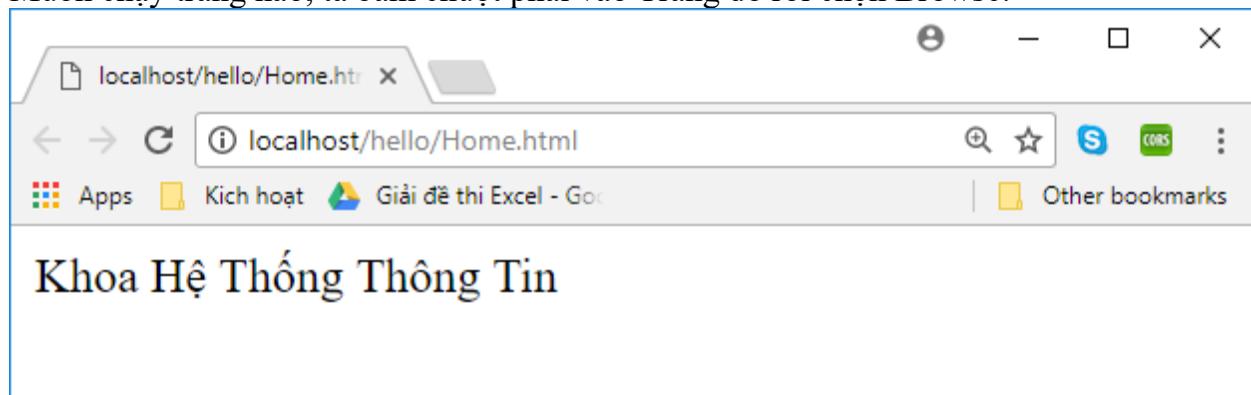
Bạn quan sát kết quả:



Ta chọn hello trong nhóm Default Web Site → chọn Tab Content View (nhìn bên dưới):

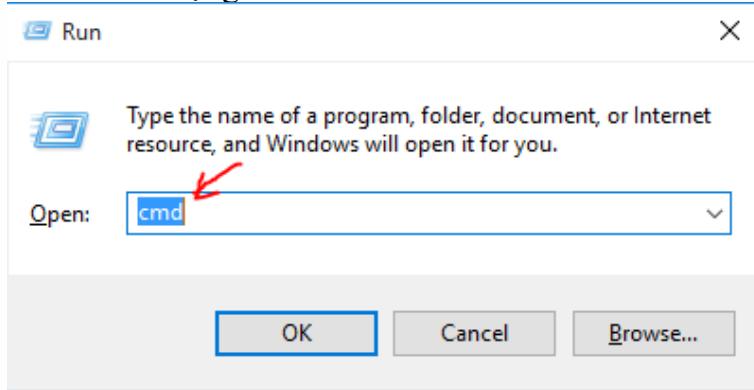


Muốn chạy trang nào, ta bấm chuột phải vào Trang đó rồi chọn Browse:



Bước 4.3: Cách lấy địa chỉ IP của máy local của ta:

– Nhấn tổ hợp phím có biểu tượng Windows + R để mở cửa sổ Run:



– Gõ lệnh **cmd**, màn hình command line hiển thị lên, tiếp tục gõ lệnh **ipconfig**

```
C:\Windows\system32\cmd.exe
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\cafe>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . :

Wireless LAN adapter Local Area Connection* 2:

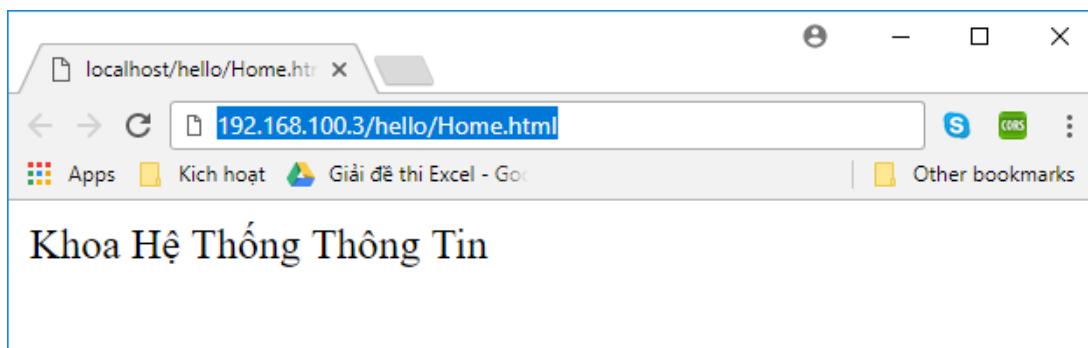
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . :

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix . :
    Link-local IPv6 Address . . . . . : fe80::e1e7:b83d:c284:c7c7%6
    IPv4 Address. . . . . : 192.168.100.3
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::1%6
                                192.168.100.1

Tunnel adapter isatap.{D17F7268-F77E-4E9F-BDA1-17A740A700F9}:
```

Ở trên Ta thấy ra địa chỉ IP **192.168.100.3** → **tùy máy của bạn nó sẽ ra khác nhau**
Bạn thay thế cho localhost → thành 192.168.100.3 → các máy tính khác sẽ truy suất vào máy của bạn theo địa chỉ IP này.



Bước 4.4: Cách cấu hình Trang Chủ mặc định cho Website

Ta nhớ rằng một Website có nhiều trang, phải có một trang mặc định làm trang chủ khi Website được vận hành → ta cần phải cấu hình. IIS Web Server mặc định lấy các Trang có tên sau làm Trang Chủ:

Default.htm

Default.asp

Default.aspx

index.htm

index.html

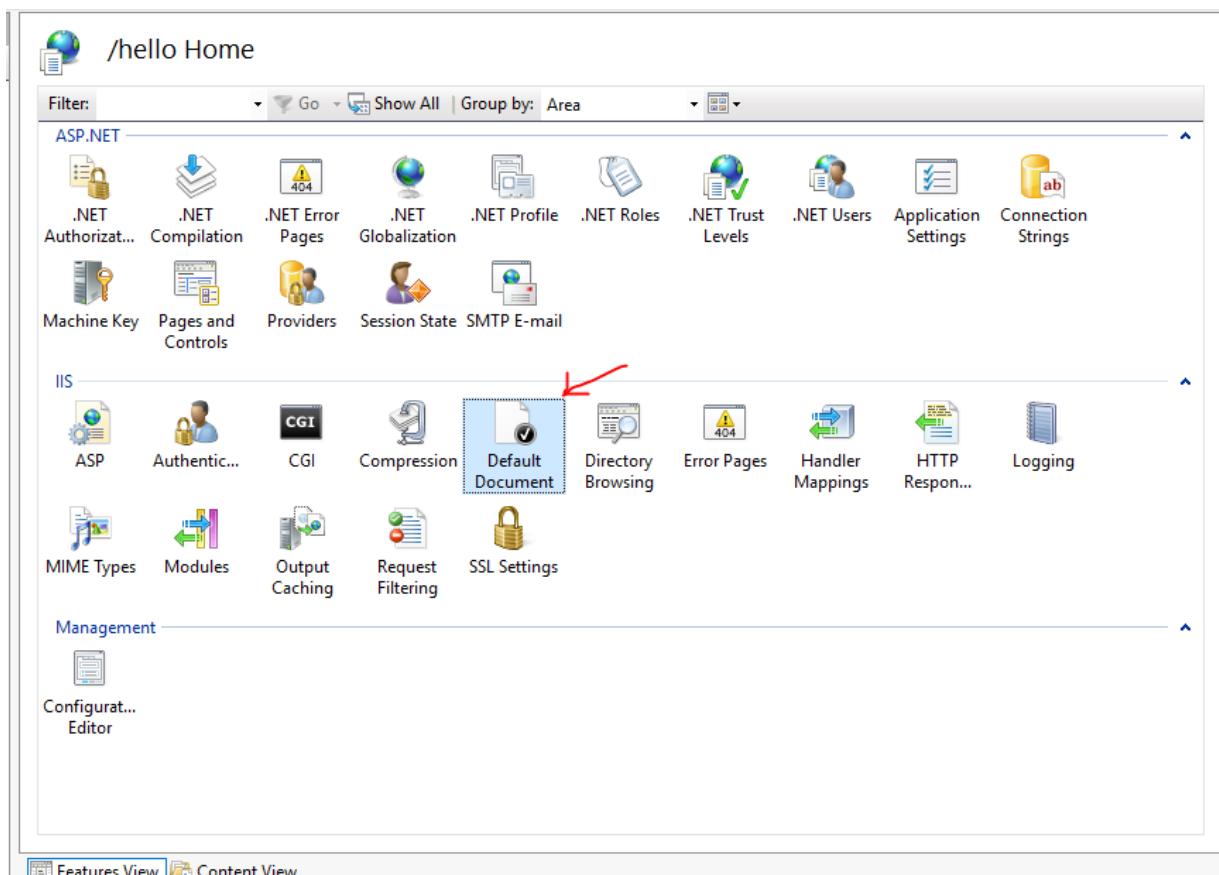
iisstart.htm

Do đó nếu trong hệ thống Website mà IIS không tìm thấy bất kỳ tên trang Web nào cùng tên ở trên thì chạy lên sẽ báo lỗi không thấy trang chủ → ta có 2 giải pháp để làm trang chủ:

Giải pháp 1: Có một Trang trùng tên với các tập tin mặc định ở trên

Giải pháp 2: Chỉ định một trang có tên bất kỳ làm trang chủ

Ta sẽ xử lý giải pháp 2 → Bạn Hello → bên phải chọn Default Document:



Bạn thấy danh sách Default Document hiển thị như dưới đây:

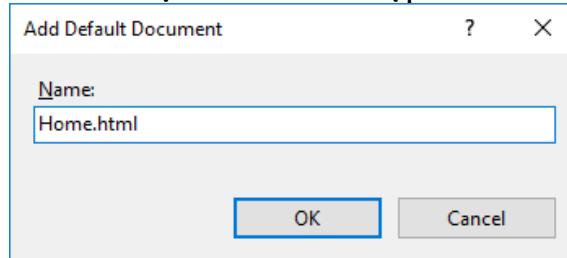
Name	Entry Type
Default.htm	Inherited
Default.asp	Inherited
index.htm	Inherited
index.html	Inherited
iisstart.htm	Inherited
default.aspx	Inherited

Bấm chuột phải chọn Add:

The context menu options are:

- Add...
- Remove
- Move Up
- Move Down
- Disable
- Revert To Parent
- Help

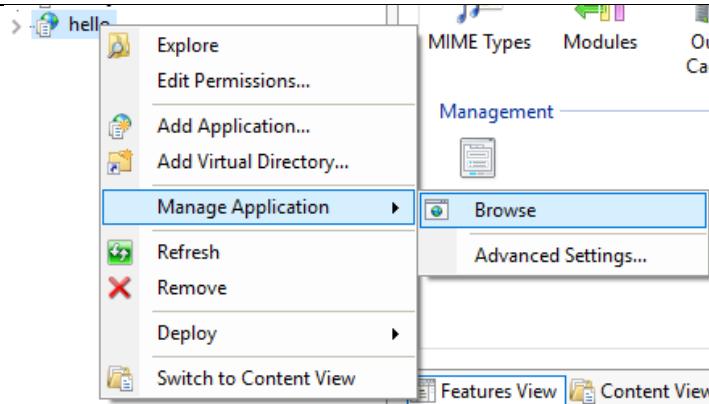
Sau đó gõ tên trang Web mà bạn muốn thiết lập nó làm trang chủ:



Bấm OK để thiết lập:

Name	Entry Type
Home.html	Local
Default.htm	Inherited
Default.asp	Inherited
index.htm	Inherited
index.html	Inherited
iisstart.htm	Inherited
default.aspx	Inherited

Sau khi thiết lập thành công, ta bấm chuột phải vào hello/chọn Manage Application/chọn Browse:



Kết quả:



Chú ý ta không còn thấy trang Home.html hiển thị trên thanh địa chỉ (nhớ rằng nếu không thiết lập trang chủ thì IIS sẽ tìm không ra khi ta chạy một Website trên IIS, nó sẽ báo lỗi). Việc thiết lập Trang Chủ là bắt buộc, ví dụ bạn có một trang Login.html → muốn trang này hiển thị ra trước thì bắt buộc phải cấu hình nó là Trang Chủ.

Bài 11: Cấu hình Hosting + Domain Free

Yêu cầu:

Hãy deploy bài tập 1 lên một Hosting miễn phí bất kỳ được giới thiệu trên lớp để bất kỳ nơi nào trên thế giới cũng truy suất được Website:

<https://azure.microsoft.com> (email education)

<https://somee.com>

<https://freevnn.com>

<https://www.000webhost.com>

<https://www.freehosting.com>

<https://www.awardspace.com/free-hosting>

<https://vhost.vn/hosting/free-hosting>

Hướng dẫn:

Sinh viên tự tìm hiểu, Nếu Somee.com có thể tham khảo tại đây:

<https://tranduythanh.com/2014/02/25/bai-44-cach-tao-webservice/>

Bài 12: Cấu hình NAT port cho Modem Internet ở nhà (*)**Yêu cầu:**

Hãy Tìm hiểu cách NAT Port cho Modem Internet ở nhà để biến máy tính cá nhân thành máy có Web Server có thể được truy suất ở bất cứ nơi nào trên thế giới: Vào IIS, Remote Desktop...

Hướng dẫn:

- Kiểm tra Modem mạng mình thuộc hãng nào, vì mỗi hãng sẽ có cách NAT port khác nhau.
- Tìm hiểu Dynamic DNS (<https://www.noip.com>)
- Tìm hiểu phần mềm Update Client for DynDN
- (<https://www.noip.com/download?page=win>)

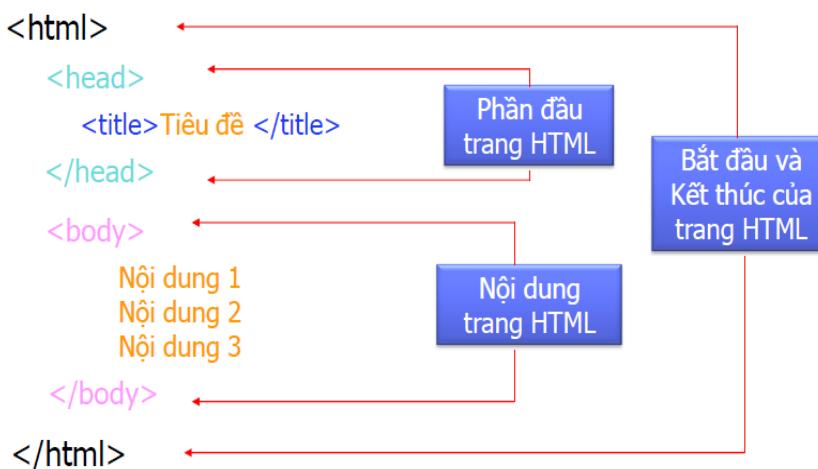
➔ Lưu ý khi cấu hình thành công, thì máy tính của ta có thể bị xâm nhập trái phép ở bất kỳ nơi đâu trên thế giới, nên phải cẩn thận bảo mật thông tin. Máy tính ta để tại nhà, nhưng ta có thể ở bất cứ vị trí nào đó trên thế giới có thể Remote vào máy, sử dụng bình thường như đang cầm máy tính trên tay. Nếu làm ở Công ty vừa và nhỏ, có Server vừa phải và để tiết kiệm nhiều chi phí đặt Server(có những dữ liệu không thể đặt nhờ) người ta có thể sử dụng giải pháp này để vận hành khai thác dữ liệu trong Công ty.

Bài 13: Cấu trúc tập tin HTML**Yêu cầu:**

Hãy trình bày cấu trúc và ý nghĩa của các thành phần chính của một Trang Web Tĩnh HTML. Giải thích tag mở, tag đóng.

Hướng dẫn:

Xem hình dưới đây để trình bày cấu trúc của một trang Web Tĩnh HTML.



Bài 14: Resume

Yêu cầu:

Tạo trang web **Resume.html**, sử dụng các tag <Hn>, , <HR>, , <A> và các tag định dạng để trình bày trang web dưới đây:

Mary Taylor

Kemper Ave Lake View co 00517(303)6455 1012



[Objectives](#) · [Education](#) · [Employment](#) · [Other Info](#) ·

Objectives

Masters degree graduate interested in a telecommunications position in the Denver or Boulder area. Highly skilled in the use of computers, audio/video equipment, and the uplink/downlink aspects of satellite communications. Interested in positions with a strong international component. Willing to Travel.

Education

Colorado State University (1999-2001)

- Graduated May: 2001.M.A. International Telecommunications
- Grade Point Average: 3.5 overall , 3.9 in major
- Dean's List : September 1999-May 2001
- Member Phi Alpha Omega Honor Society

Saint Phillip University(1996-1999)

- Graduated May,1999. B.A International Studies
- Grade Point Average:4.0 overall,4.0 in major
- Dean's List : September 1996-May 1999
- president,Honor Key Society

Employment

*Satellite Technician(Front Range Media inc.1998-1999):*Monitored satellite uplink/downlink procedures to assure quality video transmissions. Aided technicians with transmission problems. Assisted in the assembly and maintenance of uplink facility.

Technical Assistant (Mountain View Bank 1997-1998): Managed data processing system. handled user request and discussed programming options. Managed deliver service.

Salesperson (Computer Visions 1996): Sales and customer support in computers and electronics. Managed commercial accounts in Mountain View and Crabtree locations.

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai14.

<https://tranduythanh.com/webmaterials/Bai14.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 15: Beethoven

Yêu cầu:

Tạo một website **Beethoven** gồm 4 trang liên kết, mỗi trang sẽ dùng hình để liên kết đến trang sau và trang trước nó. Sử dụng các tag , <HR>, , <Hn>, <A> trình bày các trang web trong website, tự chọn hình nền cho các trang.



Beethoven's Ninth Symphony

The First Movement

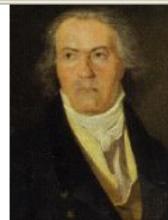
Sonata Form

1. Exposition
 1. Primary Area
 2. Transition
 3. Secondary Area
 4. Closing Area
2. Development
3. Recapitulation
 1. Primary Area
 2. Secondary Area
 3. Closing Area
4. Coda

Biểu tượng liên kết

[View the Classical Net Home Page.](#)

FirtMovement.html



Beethoven's Ninth Symphony

The Second Movement

Sonata Form

- Trang trước
1. Exposition
 2. Development
 3. Recapitulation

Trang sau

Binary Form [Trio]

1. First Half
2. Second Half
3. Scherzo Capo
4. Coda

[View the Classical Net Home page.](#)

SecondMovement.html



Beethoven's Ninth Symphony

☞ The Third Movement ☚

Sectional Form

1. A-Section
2. B-Section
3. A-Section varied
4. B-Section
5. Interlude
6. A-Section varied
7. Coda

[View the Classical Net Home page.](#)

ThirdMovement.html



Beethoven's Ninth Symphony

☞ The Fourth Movement ☚

Sonata-Concerto Form

1. Open Ritornello
2. Exposition
 1. Horror/Recitative
 2. Joy Theme
 3. Turkish Music
3. Development
4. Recapitulation
 1. Joy Theme
 2. Awe Theme
5. Codas Nos. 1 2 3

[View the Classical Net Home page.](#)

FourthMovement.html

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai15.

<https://tranduythanh.com/webmaterials/Bai15.zip>

Sinh viên tự tách dữ liệu trong tập tin ra để hiển thị đúng như yêu cầu. Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang. Biểu tượng ngón tay trỏ trang tự tìm trên mạng.

Bài 16: Euler

Yêu cầu:

Trình bày trang web **Euler.html** như hình dưới đây:

Leonhard Euler(1707-1783)

The greatest mathematician of the eighteenth century, **Leonhard Euler** was born in Basel, Switzerland. There, he studied under another giant of mathematics, Jean Bernoulli. In 1731 Euler became a professor of physics and mathematics at St. Petersburg Academy of Sciences. Euler was the most prolific mathematician of all time, publishing over 800 different books and papers. His influence was felt in physics and astronomy as well. Euler's work on mathematical analysis, *Introductio in analysin infinitorum* (1748) remained a standard textbook for well over a century. For the princess of Anhalt-Dessau he wrote *Lettres a une princesse d'Allemagne* (1768-1772), giving a clear non-technical outline of the main physical theories of the time.

One can hardly do math without copying Euler. Notations still in use today, such as e and pi, were introduced in Euler's writings. He is perhaps best known for his research into mathematical analysis. Euler's formula

$$\cos(x)+i\sin(x)=e^{ix}$$

demonstrates the relationship between algebra, complex analysis, and trigonometry. From this equation, it's easy to derive the equation

$$e^{i\pi} + 1 = 0$$

which relates the fundamental constants: 0, 1, , e, and i in a single beautiful and elegant statement.

Leonhard Euler died in 1783, leaving behind a legacy perhaps unmatched, and certainly unsurpassed, in the annals of mathematics.

Math 895: The History of Mathematics

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai16.

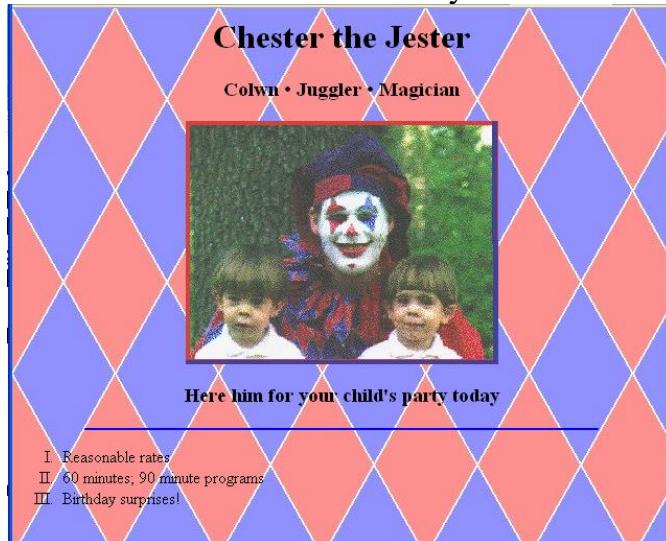
<https://tranduythanh.com/webmaterials/Bai16.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 17: Chester

Yêu cầu:

Trình bày trang web **Chester.html** như hình dưới đây:



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai17.

<https://tranduythanh.com/webmaterials/Bai17.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 18: Resume bổ sung

Yêu cầu:

Mở lại trang **Resume.html** ở bài 14 và nhập thêm nội dung sau đây vào cuối trang, sau đó tạo liên kết cho các mục:

- **References** đến trang **Reference.html**.
- **Comments on my work** liên kết đến trang **Comments.html**.
- **Go to Colorado State** link đến trang **www.colostate.edu**

Other Information

- [References](#)
- [Comments on my work](#)
- [Go to Colorado State](#)

Interested?

Contact Mary Taylor at mtaylor@tt.gr.csu.edu

Nội dung của 2 trang **Refer.html** và **Comments.html** như sau:

The screenshot shows a web page with the title "Comments about my work". At the top, there are two links: "View My Resume · References". Below the title, there are two testimonial cards. The first card is for "Lawrence Gale, Telecommunications Manager, Front Range Media Inc." It contains a quote from Karen Carlson, Manager, Mountain View Bank, praising Mary's professional skills and work ethic. The second card is for Trent Wu, Sales Manager, Computer Visions, who also praises Mary's contributions to their team. Both cards include a small red checkmark icon.

Comments about my work

View My Resume · References

Link đến trang **Resume.html** Link đến trang **Refer.html**

Lawrence Gale, Telecommunications Manager, *Front Range Media Inc.*

"Mary is a highly professional technician who takes much pride in her work. She impressed me with her ability to learn the details of our sophisticated and complex hardware and software, especially given her lack of telecommunications experience when she first started with us. Mary works well in a team but also has the ability to take my suggestions and finish a project in a highly competent manner without further direction. As she closes out her work here, I find her to be an excellent and essential component in our operations. I have complete confidence that you will be very pleased with Mary's work and recommend her very highly."

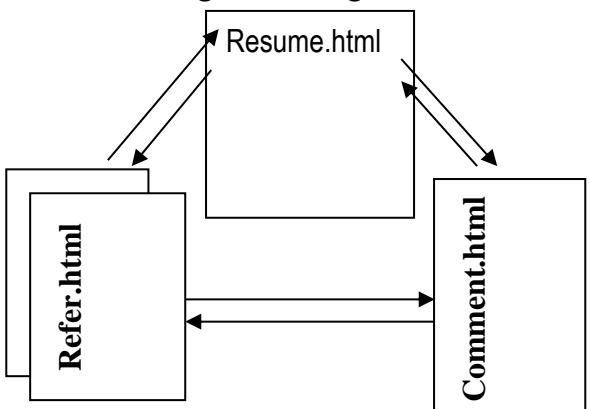
Karen Carlson, Manager, Mountain View Bank

"Mary assisted in the operations and development of a new database program we were setting up. I found Mary to be an enthusiastic and hard-working addition to our team. Mary is one of those people who gets things done and done right. She will excel in whatever she does. I think any company that hires her will be very happy that they did."

Trent Wu, Sales Manager, Computer Visions

"Mary was on our customer support staff for almost two years. During that time she never ceased to

Tạo liên kết giữa 3 trang theo mô hình:



Mở trang **Comments.html**, tạo thêm 2 liên kết đến trang **Resume.html** và trang **Refer.html**:

- [View My Resume](#) liên kết đến trang **Resume.html**
- [References](#) liên kết đến trang **Refer.html**

Mở trang **Refer.html**, tạo thêm 2 liên kết đến trang **Resume.html** và trang **Comment.html**:

References

[View My Resume](#) · [Comments](#)

Lawrence Gale, Telecommunications Manager
Front Range Media Inc.
1000 Black Canyon Drive
Fort Tompkins, CO 80517
(303) 555-0103

Karen Carlson, Manager
Mountain View Bank
2 North Maple St.
Lake View, CO 80517
(303) 555-8792

Trent Wu, Sales Manager
Computer Visions
24 Mall Road
Lake View, CO 80517
(303) 555-1313

Robert Ramirez, Prof. Electrical Engineering
Colorado State University
Kleindist Hall
Fort Collins, CO 80517

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai18.

<https://tranduythanh.com/webmaterials/Bai18.zip>

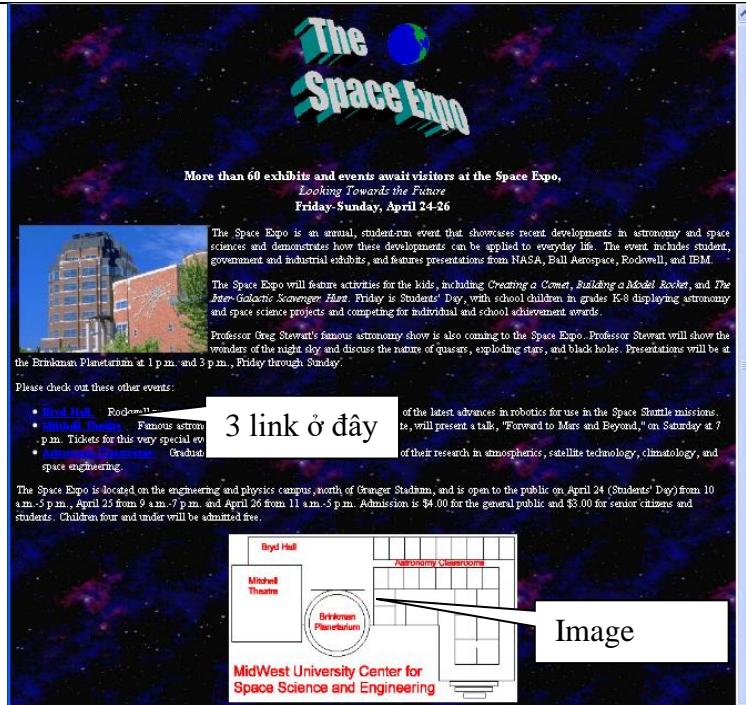
Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 19: Website Expo

Yêu cầu:

Tạo trang web **Expo.html** như hình mẫu bên dưới:

- Bryd Hall (Text và Image Maps) liên kết đến trang **Bryd.html**
- Mitchell Theatre (Text và Image Maps) liên kết đến trang **Mitchell.html**
- Astronomy Classrooms (Text và Image Maps) liên kết trang **Brinkman.html**



Hướng dẫn:

File hình, Bryd.html, Mitchell.html, Brinkman.html và dữ liệu kèm theo trong thư mục Bai19.

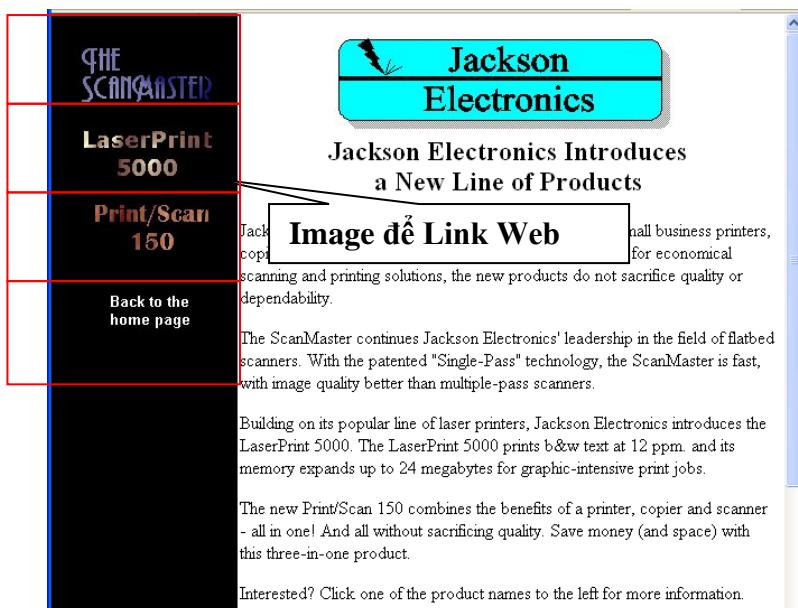
<https://tranduythanh.com/webmaterials/Bai19.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 20: Website Jaction

Yêu cầu:

Tạo trang web **Jaction.html** như hình dưới đây:



Hướng dẫn:

File hình, Scantext.htm, PStext.htm, Printext.htm và dữ liệu kèm theo trong thư mục Bai20.

<https://tranduythanh.com/webmaterials/Bai20.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 21: Website SFSF

Yêu cầu:

Tạo website **SFSF** như hình dưới đây:

The screenshot shows a website for the San Francisco Science Fiction Convention. At the top, there is a banner with the text "the San Francisco Science Fiction Convention". Below the banner, the main heading reads "Welcome to the San Francisco Science Fiction Convention". A paragraph of text provides information about the convention, mentioning it starts on Thursday, August 19th at 8 p.m. with a Get-Together party in Derleith Hall. It also mentions a costume party and a trivia contest. Below this text, there is a section for "Guests of Honor" featuring three small portraits of people. Further down, there is a registration note: "Registration is \$35 at the door, \$30 in advance. It's worth it!" followed by a link labeled "Image Maps". At the bottom, there is an address for the convention committee: "SF SF", "301 Howlitz Lane", "San Francisco, CA 94201", and a phone number "(311)555-2989".

- Image map1: link đến trang Forsttxt.htm
- Image map2: link đến trang Charntxt.htm
- Image map3: link đến trang Unwintxt.htm

Hướng dẫn:

File hình, Forsttxt.htm, Charntxt.htm, Unwintxt.htm và dữ liệu kèm theo trong thư mục Bai21.

<https://tranduythanh.com/webmaterials/Bai21.zip>

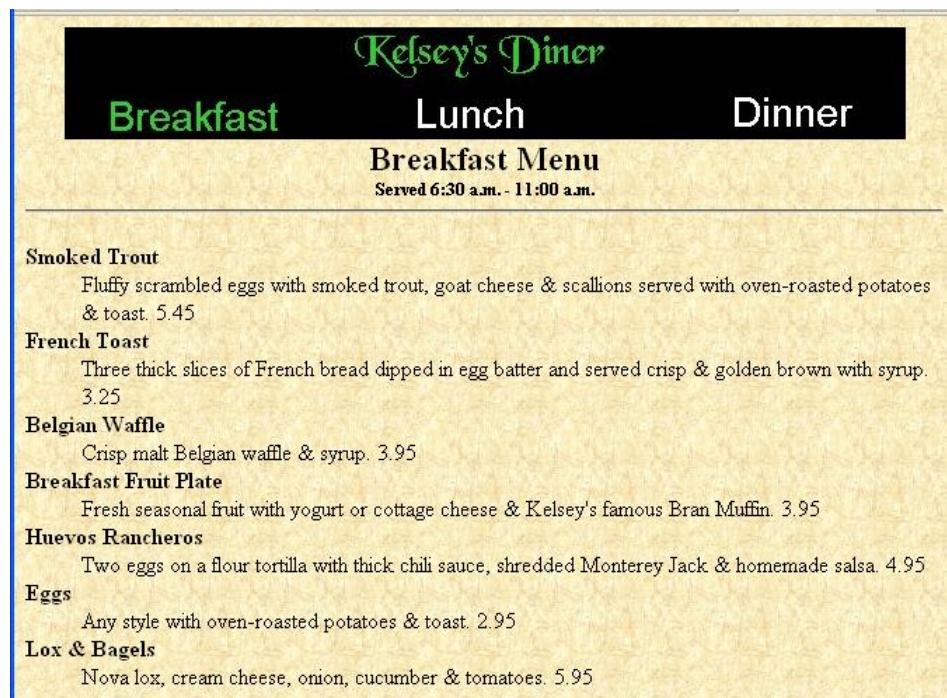
Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 22: Web Menu

Yêu cầu:

Tạo Web menu cho Kelsey's Dinner như mô tả dưới đây:

- Breakfast trỏ đến Breakfst.html
- Lunch trỏ đến Lunch.html
- Dinner trỏ đến Dinner.html



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai22.

<https://tranduythanh.com/webmaterials/Bai22.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 23: Table

Yêu cầu:

Tạo một trang web MAA.html chứa Text Tabale sử dụng tag <pre> và phím spacebar, với giao diện như mô tả dưới đây:



The Gargoyle Collection

Throughout Europe, countless gargoyles peer down from the towers and parapets of medieval cathedrals. In honor of these fascinating creations Middle Age Arts presents an exclusive line of gargoyle replicas. Choose representations from the most famous cathedrals in the world, including the popular gargoyles of Notre Dame. Select from the following list of our most popular gargoyles.

Name	Item #	Type	Finish	Price
Bacchus	48095	Wall Mount	Interior Plaster	\$95
Praying Gargoyle	48159	Garden Figure	Gothic Stone	\$125
Gargoyle	48222	Bust	Interior Plaster	\$140

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai23.

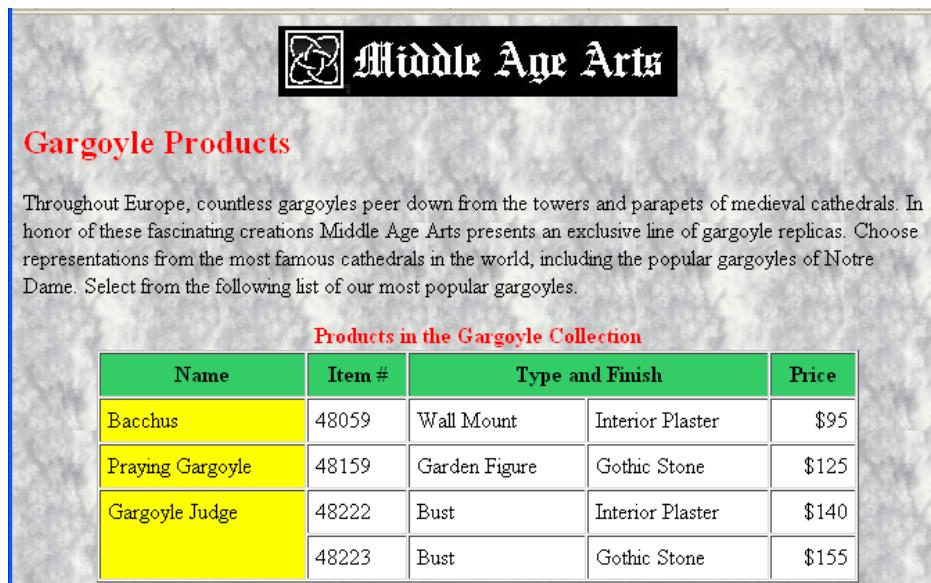
<https://tranduythanh.com/webmaterials/Bai23.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 24: Table

Yêu cầu:

Tạo một trang web MAA.html chứa với giao diện như mô tả dưới đây:



Gargoyle Products

Throughout Europe, countless gargoyles peer down from the towers and parapets of medieval cathedrals. In honor of these fascinating creations Middle Age Arts presents an exclusive line of gargoyle replicas. Choose representations from the most famous cathedrals in the world, including the popular gargoyles of Notre Dame. Select from the following list of our most popular gargoyles.

Products in the Gargoyle Collection

Name	Item #	Type and Finish	Price	
Bacchus	48059	Wall Mount	Interior Plaster	\$95
Praying Gargoyle	48159	Garden Figure	Gothic Stone	\$125
Gargoyle Judge	48222	Bust	Interior Plaster	\$140
	48223	Bust	Gothic Stone	\$155

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai24.

<https://tranduythanh.com/webmaterials/Bai24.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 25: Table Gargoyle**Yêu cầu:**

Thiết kế trang web The Gargoyle.html, sử dụng <Table> lồng nhau và các tag đã học: <A>, , <Hn>, theo mô tả dưới đây:

The screenshot displays a website for "The Gargoyle Collection". The layout is structured with a sidebar on the left and several content blocks on the right.

- Left Sidebar:** Contains links for "Home Page", "View the catalog", and "Place an order".
- Top Right Content:** Features a large image of various gargoyle statues and the text "THE GARGOYLE COLLECTION".
- Middle Left Content:** A section titled "From the President" featuring a photo of a woman and text about introducing the new Gargoyle Collection.
- Middle Right Content:** A yellow box titled "What can you do with a gargoyle?" listing items like "Bird baths", "Bookends", etc.
- Bottom Left Content:** Text describing the collection's reproductions of gargoyle figures from famous cathedrals.
- Bottom Right Content:** A section titled "Profile of the Artist" featuring a photo of Michael Cassini and text about his career and awards.

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai25.

<https://tranduythanh.com/webmaterials/Bai25.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 26: Table Dunston

Yêu cầu:

Tạo trang web **Dunston.html**, sử dụng <table> lồng nhau. Theo mô tả dưới đây:

The screenshot shows a website layout for 'the DUNSTON Retreat Center'. At the top left is a photo of a smiling man. To his right is the main logo with the text 'the DUNSTON Retreat Center' over a background image of a forest stream. Below the logo is another photo of a person in a wooded area. The page is divided into several sections: a sidebar on the left with text about the center's mission and facilities; a central column with a heading 'Next week at the Dunston Retreat Center' and details about an annual meeting; a sidebar on the right with a guest testimonial; and a bottom section with 'Upcoming Events' and specific dates for various retreats.

Welcome to the Dunston Retreat Center. Whether you are planning to attend one of our many conferences or embarking on a private retreat, we're sure that you will enjoy your stay.

Located in the northern woods of Wisconsin, the Dunston Retreat Center provides comfortable and attractive accommodations while you enjoy the rustic setting available just outside your door. The Retreat Center has 32 beds, large meeting rooms, a chapel, and kitchen facilities. If you want to get out, there are ample opportunities for hiking, canoeing and horseback riding in the surrounding area.

Throughout the year the center staff conducts retreats to accommodate the needs of various groups. We offer retreats for men, for women, and for couples. Please call about special needs retreats.

If you prefer, an individually

the DUNSTON Retreat Center

Next week at the Dunston Retreat Center

The annual meeting of the Midwest Marriage Encounter occurs at the Dunston Retreat Center, June 11-13. Registration is \$50 and includes room and board. A boating trip on Lake Superior is planned for Saturday night (\$10 fee).

Contact Maury Taylor at 555-2381 for reservation information.

Upcoming Events

June 11-13 Marriage Encounter

June 18-20 Recovering Alcoholics

June 25-27 Spirituality Workshop

A letter from one of our guests

I'm writing to tell you how much I enjoyed my retreat at Dunston. I came to your center haggard and worn out from a long illness and job difficulties. I left totally refreshed. I especially want to thank Father Thomas Holloway for his support.

I've enthusiastically told all of my friends about the wonderful place you have. Some of us are hoping to organize a group retreat. Rest assured that you'll see me again. Going to Dunston will become a ~~nearby~~ event for me.

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai26.

<https://tranduythanh.com/webmaterials/Bai26.zip>

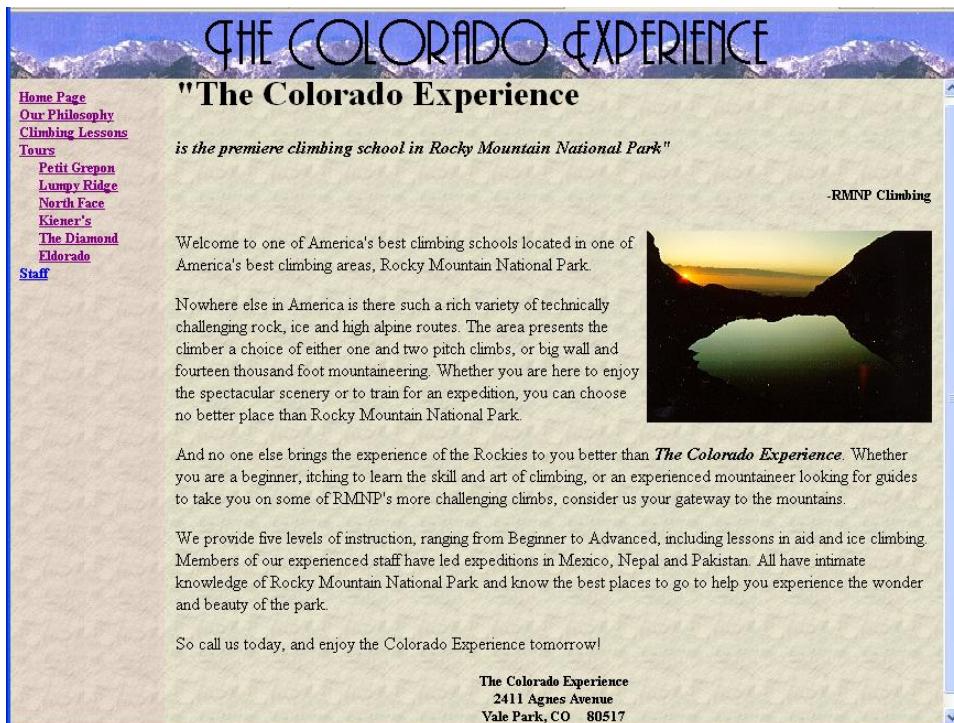
Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 27: Frame

Yêu cầu:

Tạo một website **Colorado.html** sử dụng frameset có dạng:

Logo.html	
Link.html	Homepage.html Our Philosophy.html Climbing Lessons.html Petit Grepon.html Lumpy Ridge.html North Face.html Kiener's.html The Diamond.html Eldorado.html



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai27.

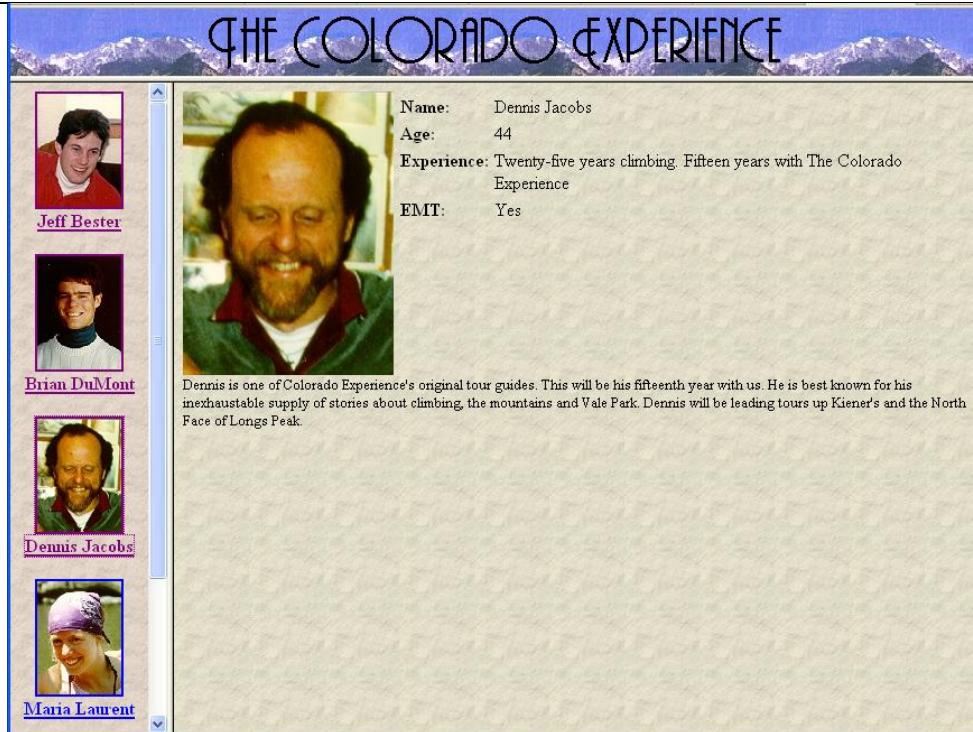
<https://tranduythanh.com/webmaterials/Bai27.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 28: Frameset

Yêu cầu:

Thiết kế website sử dụng frameset với giao diện mô tả dưới đây:



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai28.

<https://tranduythanh.com/webmaterials/Bai28.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 29: Frameset – Image maps

Yêu cầu:

Thiết kế website sử dụng Frameset và image maps, Frameset có dạng

Logo	4 trang liên kết
Image Maps	

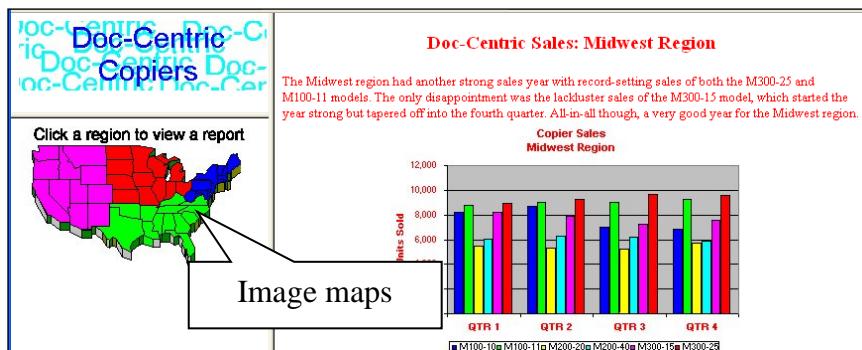


Image maps gồm 4 vùng với 4 màu, khi click vào mỗi vùng sẽ link đến một trang với màu tương ứng trong image maps.

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai29.

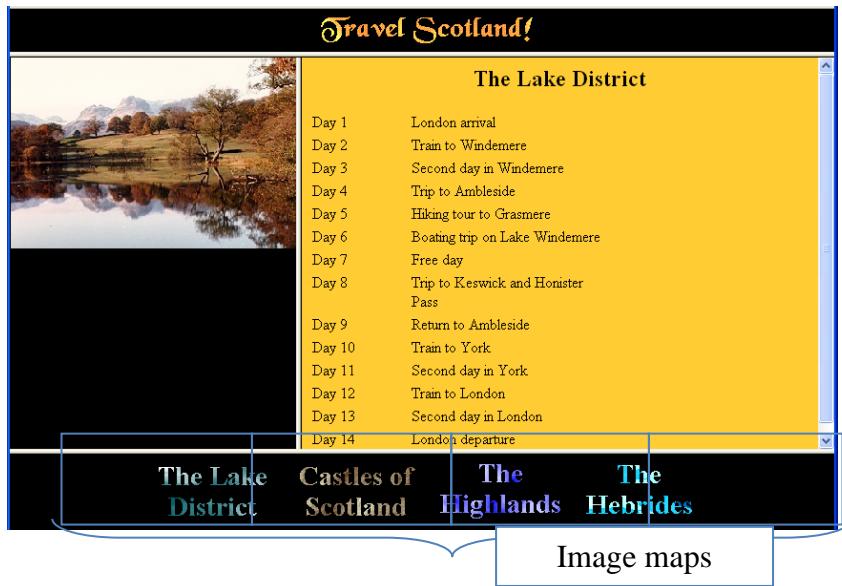
<https://tranduythanh.com/webmaterials/Bai29.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 30: Frameset – Image maps

Yêu cầu:

Thiết kế website **Travel Scotland.html**, sử dụng Frameset và image maps. Giao diện như mô tả dưới đây:



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai30.

<https://tranduythanh.com/webmaterials/Bai30.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 31: Frameset (*)

Yêu cầu:

Thiết kế website Sonnet sử dụng Frameset. Giao diện như hình dưới đây:

English 220 <i>Sixteenth and Seventeenth Century Poetry</i>	
<p>Authors</p> <p>John Donne Sonnet#1 Sonnet#5 Sonnet#10 William Shakespeare Edmund Spenser</p>	<p>Sonnet 10</p> <p>Death, be not proud, though some have called thee Mighty and dreadful, for thou art not so; For those whom thou think'st thou dost overthrow Die not, poor Death, not yet canst thou kill me.</p> <p>From rest and sleep, which but thy pictures be, Must pleasure; then from thee much more must flow, And soonest our best men with thee do go, Rest of their bones and soul's delivery.</p> <p>Thou art slave to fate, chance, kings, and desperate men, And dost with poison, war, and sickness dwell, And poppy or charms can make us sleep as well And better than thy stroke; why swell'st thou then?</p> <p>One short sleep past, we wake eternally And death shall be no more; Death thou shalt die.</p> <p><i>John Donne</i></p>

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai31.

<https://tranduythanh.com/webmaterials/Bai31.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 32:Marquee, Multimedia

Yêu cầu:

Dùng Marquee để cho dòng chữ “Khoa Hệ Thông Tin” chạy qua trái rồi qua phải màn hình Web. Chèn một bài hát làm nhạc nền cho trang này.

Hướng dẫn:

Xem Slide bài học về Marquee và Multimedia.

Module 3 – Cascading Style Sheets (CSS)

Nội dung kiến thức thực hành:

- + Năm được Khái niệm về DHTML
- + Hiểu được ý nghĩa của CSS, cách thức hoạt động, các loại CSS và quy tắc tạo CSS trong HTML
- + Năm rõ Chi tiết các thành phần thường dùng trong CSS
- + Tạo được Mô hình box
- + Hiểu được các khái niệm về Margin, Padding
- + Cách sử dụng Display, Position, Floating

Bài 33: Internal Style

Yêu cầu:

Dùng Internal Style để thiết kế Website sau:

The screenshot shows a web browser window with the following content:

- Header:** Apps, Kích hoạt, Giải đố thi Excel - Go!, Other bookmarks
- Title:** Internal Styles
- Style Properties:** background, p, strong
- Section:** Vietnam Information
 - Geography:** Inland area: 330,991 km², Continental shelf area: 1,000,000 km², Population: 78,774,000 inhabitants (in 1999), Coastline: 3,260 km, Inland coordinates: Latitude 102°01' - 109°03' East, Longitude 08°03' - 23°02' North
 - Topography:** Three quarters of Vietnam's territory consists of mountainous regions. Vietnam has two major deltas (the Red River delta and the Mekong delta) and four distinctive mountainous zones each having its own unique features.
 - People:** The Vietnamese population exceeds 78.7 million divided in 54 ethnic groups. The largest group is the Viet, or Kinh, accounting for 80% of the whole population concentrating in the plain areas while minorities occupy the highlands and the mountains.
 - Religion:** There is no official religion in Vietnam, but Buddhism is relatively popular. Some of the Catholic churches and Buddhist pagodas are interesting tourist attractions. There are religions more specific to the southern regions of the country such as the Caodaism, the Hoa Hao sect, and Islam. Most Vietnamese worship their ancestors.
- Copyright:** Copyright © BenThanh Tourist

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai33
<https://tranduythanh.com/webmaterials/Bai33.zip>

```
<style type = "text/css">
    body{
        background-image: url('hinh/camel.gif');
        background-position: right bottom;
        background-repeat: no-repeat;
        background-attachment: fixed;
    }
    p{
        font-size: 12pt;
        color: #aa5588;
        text-align: justify;
        font-family: arial, sans-serif;
    }
    strong{
        color:#0000FF;
        font-family: verdana;
    }
</style>
```

Bài 34: Inline Style

Yêu cầu:

Dùng Inline Style để thiết kế Website sau:

The screenshot shows a web browser window with the title bar 'Inline Styles'. The address bar contains the URL 'file:///F:/DaiHocKinhTeLuat/2017-2018_HKI/ThietKeWeb1/BaiTap/01/Practice01/06-l...'. The page content is as follows:

Vietnam Information

Geography

Inland area: 330,991 km²
Continental shelf area: 1,000,000 km²
Population: 78,774,000 inhabitants (in 1999)
Coastline: 3,260 km
Inland coordinates: Latitude 102°01' - 109°30' East
Longitude 08°03' - 23°02' North

Topography

Three quarters of Vietnam's territory consists of mountainous regions. Vietnam has two major deltas (the Red River delta and the Mekong delta) and four distinctive mountainous zones each having its own unique features.

People
The Vietnamese population exceeds 78.7 million, divided in 54 ethnic groups. The largest group is the Viet, or Kinh, accounting for 80% of the whole population, concentrating in the plain areas while minorities occupy the highlands and the mountains.

Religion

There is no official religion in Vietnam, but Buddhism is relatively popular. Some of the Catholic churches and Buddhist pagodas are interesting tourist attractions. There are religions more specific to the southern regions of the country such as the Caodai sect, the Hoa Hao sect, and Islam. Most Vietnamese worship their ancestors.

Copyright © BenThanh Tourist

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai34.
<https://tranduythanh.com/webmaterials/Bai34.zip>

```
<h1>Inline Styles</h1>
<p><strong>Style Properties:</strong> text, font, background, margin, padding</p>
<hr />
<h1 style="color: #AAAAAA">Vietnam Information</h1>
<p style="font-family: verdana; font-size=14pt;">
<b>Geography</b><br />
```

```
<p style="text-align: left; font-size: 14pt; background-color: #CCCCCC; padding-left: 1cm">
<b style="color: #0000FF">Topography</b><br />
```

```
<p style="margin-left: 2cm; background-image:url('hinh/camel.gif')">
<b style="color: #FF00FF">People</b><br />
```

```
<p style="margin-right: 1cm; margin-left: 1cm; text-align: justify; background-color: #CCCCCC">
<b style="color: #00FFFF; text-decoration: underline">Religion</b><br />
```

Bài 35: External Style

Yêu cầu:

Dùng External Style để thiết kế Website sau:

The screenshot shows a web browser window titled "External Styles". The address bar displays the URL: file:///F/DaiHocKinhTeLuat/2017-2018_HKI/ThietKeWeb1/BaiTap/01/Practice01/09-... . The page content is titled "External Styles" and contains a section titled "Nội dung thực hành" (Content Practice) which lists various assignments categorized by part and sub-part. To the right of the text, there is a blue silhouette of a camel.

Nội dung thực hành

I. Part 1: XHTML and CSS

1. XHTML : Basic tags, Lists, Links
 - Bài TH 01 (01-template.htm)
 - Bài TH 02 (02-Noi Dung Mon Hoc.htm)
2. XHTML : Images, Tables
 - Bài TH 01
 - Bài TH 02
 - Bài TH 03
 - Bài TH 04
3. CSS :
 - Bài TH 01
 - Bài TH 02
 - Bài TH 03
 - Bài TH 04

II. Part 2: Basic of Javascript Programming

5. Javascript - Giới thiệu về Lập trình script
 - Bài TH 01
6. Javascript - Các cấu trúc điều khiển - p1
 - Bài TH 01
7. Javascript - Các cấu trúc điều khiển - p2
 - Bài TH 01
8. Javascript - Hàm
 - Bài TH 01
9. Javascript - Mảng
10. Javascript - Đối tượng

III. Part 3: DHTML

11. DHTML - Mô hình Đối Tượng và Collections
12. DHTML - Mô hình Sự Kiện
13. DHTML - Filters và Transitions
14. DHTML - DataBinding

Copyright © 2017 -

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai35.

<https://tranduythanh.com/webmaterials/Bai35.zip>

Tạo một file tên mainstyle.css, có mã lệnh như sau:

```
body { background-image: url('hinh/camel.gif');  
       background-position: right bottom;  
       background-repeat: no-repeat;  
       background-attachment: fixed;  
     }  
p { font-size: 12pt;  
   color: #aa5588;  
   text-align: justify;  
   font-family: arial, sans-serif;  
 }  
ol {list-style-type: upper-roman;  
 }  
ol ol{list-style-type: decimal;  
 }  
ol ol ul{  
   list-style-type: circle;  
 }  
.list{  
   font-family:verdana;  
}  
a{  
   text-decoration: none  
}  
a:hover {  
   text-decoration: underline;  
   color: red;  
   background-color: #ccffcc  
}
```

Bài 36: Box Menu

Yêu cầu:

Thiết kế trang part1.html chứa hộp menu như hình bên dưới, sử dụng Internal CSS:



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai36
<https://tranduythanh.com/webmaterials/Bai36.zip>

Tạo một file tên mainstyle.css, có mã lệnh như sau:

```
h3{  
height:50px;  
width:100%;  
background:url("hinh/h3.png") no-repeat 0 100%;  
margin : 0;  
padding : 0;  
}  
ul  
{  
margin : 0;  
padding : 10px 10px 25px 45px;  
background:url("hinh/ul.png") no-repeat 0 100%;  
list-style-image:url("hinh/li.png");  
}  
/* unvisited link */  
a:link {  
color: red;  
}  
/* visited link */  
a:visited {  
color: green;  
}  
/* mouse over link */  
a:hover {  
color: hotpink;  
padding-left:10px;  
}  
/* selected link */  
a:active {  
color: blue;  
}
```

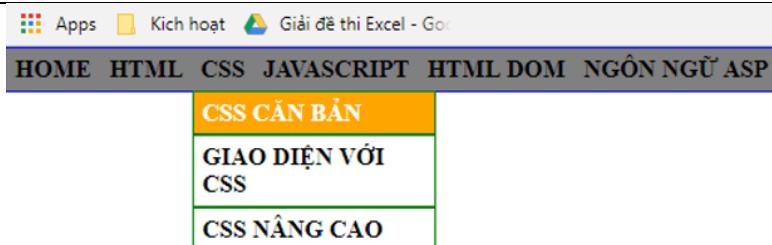
Bài 37: Menu cho Website

Yêu cầu:

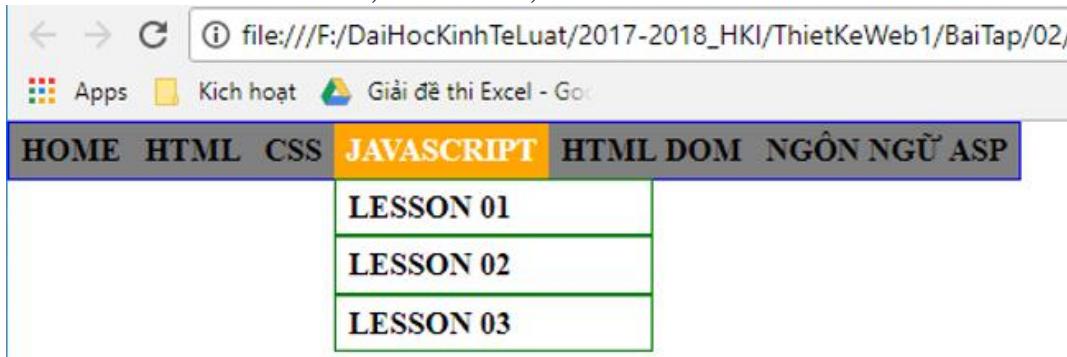
Thiết kế trang home.html, sử dụng external CSS như hình bên dưới:

Menu chính: Home, HTML, CSS, JAVASCRIPT, HTML DOM, ASP

Menu CSS: CSS Căn bản, Giao diện với CSS, CSS Nâng cao



Menu JAVASCRIPT: Lesson 01, Lesson 02, Lesson 03



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai37.

<https://tranduythanh.com/webmaterials/Bai37.zip>

Tạo một file tên mainstyle.css, có mã lệnh như sau:

```
*{  
    margin:0;  
    padding:0;  
}  
#container  
{  
    border:1px solid blue;  
    background:gray;  
    float:left;  
    top: 60px;  
    text-indent:0px;  
}  
#container ul  
{  
    float:left;  
    list-style:none;  
}  
#container ul li  
{  
    position:relative;  
    float:left;  
}  
#container ul li a  
{  
    display:block;  
    color:black;  
    font-weight:bold;  
}
```

```
text-decoration:none;
padding:.3em 6px;
}
#container li a:hover
{color:white;
background:orange;
}
#container li ul
{position:absolute;
display:none;
width:10em
}
#container li:hover ul {
display:block;
}
#container ul li ul li{
width:100%;
border:1px solid green
}
```

Bài 38: Bốn mùa

Yêu cầu:

Thiết kế trang bonmua.html, sử dụng Internal CSS như hình bên dưới.

MÙA XUÂN

Những ngày tháng Chạp, khi đất trời rộn ràng trong tinh khúc giao mùa, ta vẫn thường cùng nhau đeo qua những con phố, ngày giáp Tết đồng đúc người qua lại. Phố những ngày này chẳng xác xơ như khi mùa đông mới ghé. Phố cũng tươi tắn, xứng xinh trong những sắc màu tươi rói để chờ đón xuân sang

MÙA HẠ

Mùa mưa hạ đến rồi dì bắt chốt mang theo không khí mát dịu, trong trẻo xoa tan mảng hè đỏ lửa. Kỳ niệm ủa vú. Con mưa đầu mùa mang đến mùa mang bụi thu, nhưng có lúc lại lấy đi đồng lúa đang thi của người nông dân khắc khổ.

MÀU THU

Mùa thu tối, cái nắng ấm nhẹ nhàng của mùa thu sẽ dần đến thay đổi cái nắng oi ả mùa hè. Mùa thu đến không chỉ là bước chuyển mình của thiên nhiên mà nó còn kéo theo biết bao sự đổi khác của tâm hồn con người, một ánh nhìn mới, một xúc cảm mới và có thể là cả một sự khởi đầu mới.

MÙA ĐÔNG

Mùa đông là mùa tương hợp với những mối tình muộn màng. Mùa của những lối lầm và trán trở. Mùa mà cây sim đứng nhọc nhằn trong mưa rét nhưng không thể nở hoa. Đó cũng là mùa ủ áp những lộc chồi. Mùa xuân bắt đầu bằng những cơn gió lạnh đến tai té từ lúc cuối đông...

Hướng dẫn:

Tạo Tag Style định dạng một số thuộc tính cho thẻ p

```
width: 45%;  
margin-top:...;  
margin-left:...;  
float:left;
```

Nhập dữ liệu cho đoạn 1 bình thường cuối đoạn nhấn phím Enter để thấy kết quả. Tương tự nhập cho các đoạn tiếp theo.

Tạo lớp định dạng cho tiêu đề

```
font-size: ...;  
font-weight: ...;  
color: #00F;
```

Tạo ID định dạng màu nền đặc trưng cho 4 mùa

```
background-color:...;  
color:...;
```

File hình và dữ liệu kèm theo trong thư mục Bai38.

<https://tranduythanh.com/webmaterials/Bai38.zip>

Bài 39: Table with external CSS

Yêu cầu:

Thiết kế trang fantastable .html như hình bên dưới, sử dụng external CSS:

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Berglunds snabbköp	Christina Berglund	Sweden
Centro comercial Moctezuma	Francisco Chang	Mexico
Ernst Handel	Roland Mendel	Austria
Island Trading	Helen Bennett	UK
Königlich Essen	Philip Cramer	Germany
Laughing Bacchus Winecellars	Yoshi Tannamuri	Canada
Magazzini Alimentari Riuniti	Giovanni Rovelli	Italy
North/South	Simon Crowther	UK
Paris spécialités	Marie Bertrand	France

Hướng dẫn:

```
#customers{  
font-family:....;  
width:100%;  
border-collapse:collapse;  
}  
#customers td, #customers th  
{  
font-size:....;  
border:1px solid #98bf21;  
padding:3px 7px 2px 7px;  
}  
#customers th  
{  
font-size:1.1em;  
text-align:left;  
padding-top:5px;  
padding-bottom:4px;  
background-color:#A7C942;  
color:#ffffff;  
}  
#customers tr.alt td  
{  
Color:....;  
background-color:....;  
}
```

Bài 40: Table with external CSS

Yêu cầu:

Thiết kế trang *tablecss.html* như hình bên dưới, sử dụng external CSS:

Name	Phone	E-Mail
Catherine Adler Library Director	555-3100	
Michael Li Head of Adult Services	555-3145	
Kate Howard Head of Technical Services	555-4389	
Robert Hope Head of Children's Services	555-7811	
Wayne Lewis Circulation Services Supervisor	555-9001	
Bill Forth Interlibrary Loan	555-9391	

Hướng dẫn:

padding:...

float:...

margin-bottom:...

margin-left:...

border: ...

Bài 41: Vietnam Airline Website – external CSS

Yêu cầu:

Thiết kế trang *vnairlines.html* như hình bên dưới, Sử dụng External CSS:

Vé điện tử là gì?

"Vé hành khách": là chứng từ vận chuyển hành khách và hành lý bằng đường hàng không và là bằng chứng của việc giao kết hợp đồng, các điều kiện của hợp đồng. Vé hành khách được thể hiện dưới hai hình thức là Vé giấy (còn thể hiện dưới tên Vé hành khách và hành lý) và Vé điện tử.

"Vé điện tử": là vé hành khách được thể hiện dưới dạng dữ liệu điện tử, bao gồm các thông tin liên quan đến việc sử dụng dịch vụ vận chuyển của Vietnam Airlines (HKVN) thay thế cho việc xuất Vé giấy. Vé điện tử có đầy đủ các thông tin như trên Vé giấy của HKVN và HKVN có trách nhiệm lưu trữ Vé điện tử và đảm bảo quá trình di lại của hành khách được thuận tiện.

"Hợp đồng vận chuyển hành khách, hành lý": là bộ tài liệu bao gồm Vé điện tử, Điều lệ vận chuyển của HKVN, bảng giá cước vận chuyển và các thoả thuận khác bằng văn bản giữa hai bên.

Tiền lợi của vé điện tử

- An toàn hơn cho quý khách vì tránh được rủi ro mất, rách, nát.
- Giúp quý khách làm thủ tục chuyến bay nhanh chóng hơn.
- Các thay đổi liên quan đến Vé điện tử được thực hiện dễ dàng hơn do nhân viên có thể tra cứu thông tin nhanh và chính xác hơn.
- Thuận lợi trong giao dịch giữa hãng hàng không và quý khách khi điều kiện cho phép bán qua mạng Internet.
- Trong tương lai, khi điều kiện kỹ thuật cho phép, quý khách có thể tự làm thủ tục chuyến bay (check-in) để tiết kiệm thời gian.

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai41.

<https://tranduythanh.com/webmaterials/Bai41.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 42: Maxwell Scientific

Yêu cầu:

Dùng style sheet để thiết kế website với Giao diện tương tự như minh họa:

The screenshot shows the homepage of Maxwell Scientific. At the top, there's a navigation bar with links for Astronomy, Chemistry, Electronics, Engineering, and Physics. Below the navigation is a main content area. On the left, a section titled "Astronomy" features a list of "Featured Astronomical Products". This list includes items like a Refractor Kit, NightDisk, Constellation Globe, Star and Planet Locators, Rechargeable Red Flashlight, Classroom Planetarium, Solar Mobile, and Solar Mobile Kit. To the right of this list is a yellow-bordered box titled "Run the Messier Marathon" which contains text about the annual Messier Marathon event. At the bottom of the page, there's a section titled "Comments From Our Astronomy Customers" with three testimonial snippets from happy customers. The footer contains the company address: 205 East Nordheim Drive, Hapton, IL 43129, and phone number: (800) 555-2191 Fax: (812) 555-2192.

Hướng dẫn:

- Dữ liệu và hình ảnh trong thư mục bai42.

<https://tranduythanh.com/webmaterials/Bai42.zip>

– Tập tin style sheet:

```
BODY {  
    color:green;  
    background: white url(Draft.jpg) no-repeat fixed center center  
}  
H1, H2, H3, H4, H5, H6 {  
    font-family: Arial, Helvetica, sans-serif  
}  
ADDRESS {  
    font-size: 0.9em;  
    font-style:normal;  
    text-align:center;  
    text-transform:uppercase  
}  
BLOCKQUOTE{  
    background-color:silver  
}  
UL {list-style:circle url(Apple.jpg) outside}  
UL B {  
    color:rgb(155,0,0)  
}  
A {  
    font-size: 0.9em;  
    color:green  
}  
A:hover {  
    color:red;  
    text-transform:uppercase;  
    font-weight:bold  
}  
.Special {  
    color: rgb(153,102,6);  
    font-weight:bold  
}  
DIV.Article {  
    padding: 0.5em;  
    border-style: solid;  
    border-width: 2px;  
    background-color: rgb(252,221,163);  
    width: 250px;  
    float:right  
}
```

Bài 43: StuffShop

Yêu cầu:

Dùng style sheet thiết kế trang web **StuffShop** như hình bên dưới.

The Stuff Shop is your online home for the buying, selling, and trading of used merchandise and unique collectibles.

Click a link on the left to browse the store.

Antiques
Books
Clothes
Electronics
Furniture
Jewelry
Music and Videos
Sporting Goods

Hướng dẫn:

- Dữ liệu và hình trong thư mục bai43.

<https://tranduythanh.com/webmaterials/Bai43.zip>

Bài 44: M.Lee's

Yêu cầu:

Dùng style sheet thiết kế trang web **M.Lee's** như hình bên dưới:

The template features a large image of a person performing a high kick. The title "TAE KWON DO" is prominently displayed in the center. To the right, there is a green box containing a bulleted list of services. Below the main image, there is descriptive text about the classes offered, followed by a contact address at the bottom.

- 30 Years of Experience
- Day and Evening Classes
- Group and Private Lessons
- Family Rates and Classes
- Children's Classes
- Radio kick Boxing

We are proud to offer special classes for children, starting as young as 5 years old. Tae Kwon Do teaches our youth confidence, discipline and self-control. Children's classes start every day at 3 p.m. After school pick-up is available.

MASTER LEE'S TAE KWON DO
211 OAKVIEW LANE GREENDALE, IL 60111 (414) 555-2891

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai44.

<https://tranduythanh.com/webmaterials/Bai44.zip>

Bài 45: ScrapBooks

Yêu cầu:

Dùng style sheet thiết kế trang web **ScrapBooks** như hình bên dưới:

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai45.

<https://tranduythanh.com/webmaterials/Bai45.zip>

Bài 46: Mount Rainier News

Yêu cầu:

Dùng style sheet thiết kế trang web như hình bên dưới, sinh viên tự thiết kế:

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai46.

<https://tranduythanh.com/webmaterials/Bai46.zip>

Bài 47: Table with Style-trigger**Yêu cầu:**

Dùng style sheet thiết kế trang web như hình bên dưới, sinh viên tự thiết kế:

STT	Mã	Tên
1	K174111253	Trần Thị Yến Linh
2	K174111260	Trần Trọng Nghiêm
3	K164060845	Nguyễn Khiết Thành Huy
4	K164060854	Trịnh Thị Hồng Vân
5	K184111428	Nguyễn Thị Tiên

Designed by Tèo Đẹp Trai

yêu cầu tách cấu trúc của table ra làm 3 phần rõ rệt: thead, tbody, tfoot

thead là dòng tiêu đề

tbody là 5 dòng dữ liệu

tfoot là dòng cuối cùng

Định dạng CSS như trên.

Đồng thời lưu ý dòng lẻ tô màu whitesmoke, dòng chẵn tô màu pink (và phải tự động tô màu, khi có thêm dòng mới nó phải tự động nhận dạng đâu là dòng chẵn đâu là dòng lẻ để tô nền). Di chuyển chuột vào dòng nào thì dòng đó tô nền vàng và con trỏ chuột có hình ngón trỏ tay.

Hướng dẫn:

Cấu trúc table tự làm, cấu trúc style (dùng internal hoặc external style để làm).

```
<style type="text/css">
    thead, tfoot
    {
        background-color:blue;
        color:white;
    }
    tbody tr:nth-child(2n+1)
    {
        background-color:whitesmoke
    }
    tbody tr:nth-child(2n) {
        background-color: pink
    }
    tbody tr:hover
    {
        background-color:yellow;
        cursor:pointer
    }
</style>
```

Module 4 – Form và các Control trên Form

Nội dung kiến thức thực hành:

- + Hiểu được ý nghĩa và chức năng của Form
- + Phân biệt và thực hành được các phương thức GET/POST/PUT/DELETE
- + Hiểu và sử dụng được các control trên Form: Label, Textbox, Button, Combo, ListBox, TextArea....

Bài 48: Register

Yêu cầu:

Tạo trang **Register.html**, Giao diện như mô tả dưới đây:

The screenshot shows a registration form for Jackson Electronics. At the top, there's a logo with a lightning bolt icon and the text "Jackson Electronics". Below the logo, the title "Registration Form" is displayed. A sub-instruction "Register your Jackson Electronics product here" is present. The form consists of several input fields:

- First Name: Textbox
- Last Name: Textbox
- Address #1: Textbox
- Address #2: Textbox
- City: Textbox
- State: Textbox
- Zip: Textbox
- Country: Textbox (set to "United States")
- Product: Dropdown menu (selected value is "ScanMaster")
- Date Purchased: Textbox
- Serial #: Textbox
- Used for: Radio buttons (Business is selected)
- System (check all that apply):
 - Windows: Checkbox
 - UNIX: Checkbox
 - Macintosh: Checkbox
 - Other: Checkbox
- Comments?: Textarea

At the bottom of the form are two buttons: "Click to register" and "Click to cancel".

Click to register → alert thông báo Đăng ký, Click to cancel → Xóa trống dữ liệu trên Form

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai48.

<https://tranduythanh.com/webmaterials/Bai48.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 49: Online Classified

Yêu cầu:

Thiết kế trang form **Online Classified** theo yêu cầu như dưới đây:

Online Classifieds Search Form

1) Search for a classified in the following sections:

- Employment Personal
 For Sale Miscellaneous
 Housing All of the above

2) Search the following publications:

- Midwest times Modern News Employment Today
 Great Lakes Classifieds Middleton Daily Cashtown Daily News

3) From to mm/yy to dd/yy.

4) Enter a few keywords to describe the classified.

5) Click one of the options below:

 **Search**  **Help**  **Cancel**

Giả lập:

Search → giả lập Search

Help → hiển thị 1 trang help(Sinh viên tùy chọn)

Cancel → xóa trắng dữ liệu trên Form

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai49.

<https://tranduythanh.com/webmaterials/Bai49.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 50: Travel Expense Report

Yêu cầu:

Thiết kế trang form như hình bên dưới:

The screenshot shows a web-based travel expense report form. At the top, there is a logo for 'DeLong Enterprises' on a brown background. Below the logo, the title 'Travel Expense Report' is displayed in bold black font. The form consists of several input fields and a table for expense items. The fields include:
1) First Name: [text box] Last Name: [text box]
2) Social Security Number: [text box]
3) Department: [dropdown menu set to Accounting]
4) Describe the purpose of your trip.
[Large text area]
5) Itemize your travel expenses
[Table]
The table has four columns: Date, Description, Category, and Amount. It contains four rows, each with a date field ('mm/dd/yy'), a description field (empty), a category dropdown ('Meal'), and an amount field (empty).
6) Have you submitted your receipts? [radio button set to Yes]
[Submit travel expenses] [Reset]
Questions? Contact Debbie Larson in Accounting (ext. 2150).

Submit → Thông báo submit

Reset → Xóa trắng ô ban đầu

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai50.

<https://tranduythanh.com/webmaterials/Bai50.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 51: Registration**Yêu cầu:**

Thiết kế trang form **Registration**, Giao diện như hình dưới đây:

The screenshot shows a registration form for newborns. At the top, it says "St. Mary's of Northland Pines" and "Research Hospital and Clinics". The main title is "Newborn Registration".
Fields include:
1) Infant's first name: [text box] Last: [text box]
2) Medical Record #: [text box]
3) Date of birth: mm/dd/yy
4) Birth weight: [text box]
5) APGAR score: [text box]
6) Mother's first name: [text box] Last: [text box]
7) Father's first name: [text box] Last: [text box]
8) Physician (check all that apply):
Dr. Warren Albert
Dr. Maria Alvarez
Dr. Karen Brinkman
Dr. Micheal Kerry
Buttons at the bottom: Register, Cancel

Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai51.

<https://tranduythanh.com/webmaterials/Bai51.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

Bài 52: Contact - GET**Yêu cầu:**

Tạo trang liên hệ như hình dưới đây:

Họ và tên :

Email :

Giới tính : Nam, Nữ

Tôi đã có nghề nghiệp ổn định

Đến từ

Nội dung câu hỏi chúng tôi :

Nội dung trả lời

Miền Bắc
Miền Trung
Miền Nam
Miền Tây

Khi bấm gởi thông điệp → dùng Method GET để chuyển toàn bộ dữ liệu trong trang Liên hệ ra một trang mới (Sinh viên tự thiết kế).

Hướng dẫn:

Tham khảo bài lấy dữ liệu trong lớp lý thuyết

Bài 53: Sản phẩm-GET

Yêu cầu:

Tạo trang web **Danh sách sản phẩm dưới đây, kết hợp Table + TextBox, Button**

Part Number	Description	Price	Qty	Item table
C112	Chuột Quang	19.95\$	<input type="text"/>	<input type="text"/>
B124	Bàn Phím	29.95\$	<input type="text"/>	<input type="text"/>
U125	USB KingMax	39.95\$	<input type="text"/>	<input type="text"/>

Total

Dùng Method GET để xử lý nút TongCong → gửi toàn bộ dữ liệu qua một trang HTML mới (Sinh viên tự nghĩ ra trang HTML này)

Hướng dẫn:

Tham khảo bài lấy dữ liệu trong lớp lý thuyết

Module 5 – Javascript

Nội dung kiến thức thực hành:

- + Hiểu ngôn ngữ JavaScript trong thiết kế và lập trình web
- + Biết cách sử dụng các qui tắc và cú pháp của ngôn ngữ HTML vào việc hỗ trợ thiết kế giao diện và thiết kế xử lý của trang web
- + Biết cách gắn kết giữa ngôn ngữ JavaScript, các hàm JavaScript với các thẻ HTML
- + Nắm vững quy tắc và kỹ thuật xử lý chuỗi bằng Javascript
- + Xây dựng và sử dụng được các lớp đối tượng trong JavaScript

Bài 54: Hàm toán học

Yêu cầu:

Tạo trang web với Table và Textbox, Button để tính MAX, MIN, sin, cos... Theo mô tả dưới đây:

a(So Thu 1):	30
b(So Thu 2)	45
c(So Thu 3)	90
KET QUA	-0.9880316240928618
MAX	MIN
sin(a)	Cos(a)
a^b	

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

Bài 55: Giải phương trình bậc 2

Yêu cầu:

Viết WebSite giải phương trình bậc 2 theo mô tả dưới đây:

Giải phương trình bậc 2

Hệ số a:	1
Hệ số b:	-2
Hệ số c:	1
Giải	Tiếp
Kết quả:	Nghiệm kép $x_1=x_2=1$

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

Bài 56: Xử lý chuỗi**Yêu cầu:**

Tạo trang web Xử lý chuỗi Theo mô tả dưới đây (Dùng table và các control):

Sử dụng các hàm Xử Lý Chuỗi	
Nhập dữ liệu	
<input type="text"/>	
<input type="button" value="Nhập dữ liệu"/>	<input type="button" value="Đếm ký tự hoa"/>
<input type="button" value="In chữ Hoa"/>	<input type="button" value="In mỗi từ trên mỗi dòng"/>
<input type="button" value="In chữ thường"/>	<input type="button" value="Đếm số từ"/>
<input type="button" value="Đếm số ký tự thường"/>	<input type="button" value="In Nguyễn âm, phụ âm"/>
<input type="button" value="Xem Website xử lý chuỗi w3c"/>	

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

Bài 57: Xử lý sự kiện text change**Yêu cầu:**

Tạo trang web với Table và các control. Theo mô tả dưới đây:

Mã sản phẩm	Tên Sản phẩm	Giá	Số lượng	Thành tiền
C112	Chuột Quang	10\$	<input type="text" value="2"/>	<input type="text" value="20\$"/>
B124	Bàn Phím	20\$	<input type="text" value="3"/>	<input type="text" value="60\$"/>
U125	USB KingMax	15\$	<input type="text" value="1"/>	<input type="text" value="15\$"/>
Total			<input type="text" value="95\$"/>	<input type="button" value="TongCong"/>

- Người dùng nhập số lượng vào các Textbox số lượng trong mỗi dòng → Ô thành tiền sẽ tự động tính (Giá * số lượng). Yêu cầu kiểm tra tính hợp lệ dữ liệu
- Bấm vào Tổng Cộng → hiển thị tổng tiền vào ô Total

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

Bài 58: Neonatal Feeding Study**Yêu cầu:**

Thiết kế trang form nhu hình, với các yêu cầu như sau:

- Khi load form thì Today is sẽ hiển thị ngày hiện hành theo định dạng d/m/yyyy và dấu nháy nhảy đến name.
- Trong mục Physician, nếu người dùng chọn other thì khi đưa ra ngoài sẽ xuất hiện thông báo yêu cầu nhập Name.
- Nhập giá trị vào các ô Activity, Pulse, Grimace, Appearance và Respiration, chương trình kiểm tra giá trị nhập từ 0→2, nếu sai chương trình hiện thị thông báo nhập lại. Khi nhập giá trị trong từng ô thì ô Total sẽ tính tổng giá trị các ô đã nhập.

The screenshot shows a web form titled "Neonatal Feeding Study" under the header "St. Mary's of Northland Pines Research Hospital and Clinics". The form includes fields for Name, Medical Record #, Date of birth, Physician (with a dropdown menu showing "Dr. Warren Albert"), and a field for "If other (specify)". It also contains a section for "5) 1 Minute APGAR Score" with five input fields for Activity, Pulse, Grimace (Reflex, Irritability), Appearance (Skin Color), and Respiration, each with a value of 0. A "TOTAL" field is shown with a value of 0. Below this, there is a field for "6) Birth weight (gms)" with a note that weights must be ≥ 1200 gms, and a checkbox for "7) Parental Consent (required)". At the bottom are "Register" and "Reload Page" buttons.

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

File hình và dữ liệu kèm theo trong thư mục Bai58.

<https://tranduythanh.com/webmaterials/Bai58.zip>

Bài 59: North Pole Novel Ties

Yêu cầu:

Thiết kế trang web **NPN.htm**, với các yêu cầu sau:

- Dùng style sheet để định dạng các header, và các link không có gạch chân, khi đưa chuột ngang thì các link chuyển sang màu đỏ.
- Dùng Javascript viết hàm tính số ngày còn lại tính từ ngày hiện hành của máy đến ngày Noel (nhìn chõ nền vàng).



Hướng dẫn:

File hình và dữ liệu kèm theo trong thư mục Bai59.

<https://tranduythanh.com/webmaterials/Bai59.zip>

Ngoài ra sinh viên có thể sử dụng hình để làm nền cho trang.

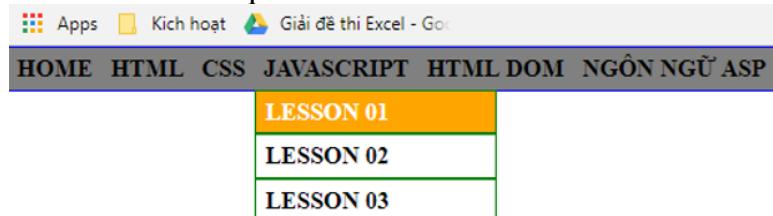
Bài 60: Menu

Yêu cầu:

Thiết kế trang web có Menu với yêu cầu dưới đây, kết hợp CSS và Javascript:
Khi di chuyển vào Menu CSS → Xô ra 3 Menu con bên dưới:



Khi di chuyển vào Menu JavaScript → Xổ ra 3 Menu con bên dưới



Di chuyển tới bất kỳ mục Menu nào đều đổi màu nền thành màu Cam như trên hình.

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý. Và tham khảo thêm trên Google.

Bài 61: Color Picker

Yêu cầu:

Thiết kế trang Color Picker như hình bên dưới:

A Simple Color Picker

Background (enter values from '00' to 'FF')
RED: FF GREEN: FF BLUE: FF

Text (enter values from '00' to 'FF')
RED: 00 GREEN: 00 BLUE: 00

Apply Reload

HAMLET:
Speak the speech, I pray you, as I pronounced it to you, trippingly on the tongue: but if you mouth it, as many of your players do, I had as lief the town-crier spoke my lines.

*If you are using Netscape, you won't be able to change the text color.

Bấm Apply sẽ áp dụng màu cho nội dung bên phải

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý. Và tham khảo thêm trên Google.

Module 6 – XML - Mô hình DOM và JSON

Nội dung kiến thức thực hành:

- + Hiểu về HTML DOM trong thiết kế web
- + Nắm được những tính năng ưu việt của HTML DOM kết hợp với JavaScript
- + Vận dụng tính động của HTML DOM vào việc thiết kế và xây dựng website
- + Hiểu và phân tích được cấu trúc tài liệu XML
- + Hiểu được cơ chế hoạt động của AJAX trong việc truy suất dữ liệu
- + Biết được cấu trúc dữ liệu JSON
- + Khai báo và khai thác sử dụng được JSON

Bài 62: Table Mouse Trigger

Yêu cầu:

Viết lệnh di chuyển chuột vào dòng nào trong Table thì tô nền vàng dòng đó, di chuyển chuột ra thì tô nền xanh.

STT	Tên
1	Nguyễn Thị Tú
2	Trần Văn Kỳ
3	Hồ Văn Năng

Hướng dẫn:

Dùng Javascript để xử lý cho các TR, mỗi TR sẽ có 2 sự kiện Mouse Over và Mouse Leave, mỗi một hàm nên viết hàm riêng để xử lý.

Bài 63: Đăng ký thành viên

Yêu cầu:

Tạo trang web **Đăng ký thành viên**, sử dụng <table> và các control trên Form, đối tượng trong JavaScript. Theo mô tả dưới đây:

Đăng ký thành viên

Tên(*)

Ngày sinh(*) Ngày: Tháng: Năm:

Giới tính(*) Nam Nữ

Email(*)

Sở thích của bạn:

- Mua sắm
- Du lịch
- Tán gẫu
- Đọc sách
- Nghe nhạc

Màu yêu thích của bạn:

- Mau xanh da trời
- Mau đỏ
- Mau vàng
- Mau xanh xanh lá cây
- Mau xanh tím

Tên	Email	Giới tính	Năm sinh	Sở thích	Màu yêu thích
Nguyễn Văn Bình	binh@gmail.com	Nam	02/02/1990	đọc sách	Mau vàng
Nguyễn Thị An	nguyenthian@gmail.com	Nữ	01/01/1992	Tán gẫu, đọc sách	Mau đỏ

- Ngày, tháng, năm sinh: Viết javascript tự động tải lên dropdown(combo), dùng các loại vòng lặp để nạp dữ liệu cho control.
- Khi nhấn đăng ký: Kiểm tra Tên không được bỏ trống, Email phải hợp lệ. Mọi thông tin nhập trên Form sẽ được hiển thị vào Table ở bên dưới
- Bấm Tiếp: Xóa dữ liệu cũ trên Form, focus tới ô Tên.
- Bổ sung sự kiện mouse over và mouse out cho các dòng trong Table. Khi di chuyển chuột vào thì nền màu vàng, di chuyển chuột ra nền màu trắng.

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

Bài 64: Dynamic Table

Yêu cầu:

Thiết kế Website tương tác Table như hình minh họa dưới đây:

23:59:04

Click chuột vào dòng này. Sẽ xuất hiện thông báo phím bấm của chuột

Thêm dòng mới ở đây	Thêm dòng mới ở đây
Thêm dòng mới ở đây	Thêm dòng mới ở đây
Thêm dòng mới ở đây	Thêm dòng mới ở đây
Dòng 1 cột 1	Dòng 1 cột 2
Dòng 2 cột 1	Dòng 2 cột 2
Dòng 3 cột 1	Dòng 3 cột 2

Thêm vào **Xoá** **Chỉ hiện dòng** **Chỉ hiện cột** **Mở một trang mới**

- Giờ phút giây ở bên trên Website → Hiển thị dạng giờ điện tử, lấy đúng giờ hệ thống của máy tính client
- Dòng Click chuột vào dòng này → Click trái thông báo là chuột trái, click phải thông báo là chuột phải (dùng alert để thông báo)
- Đối với Table:
 - Nhấn chuột vào dòng nào thì tô nền đỏ dòng đó, nhấn lại một lần nữa thì bỏ tô nền đỏ
 - Thêm vào: Chèn một TR vào Table
 - Xóa: Dòng nào có nền đỏ thì xóa hết
 - Chỉ hiện thị dòng, Chỉ hiển thị cột
 - Mở một trang mới: Viết lệnh mở trang <https://tranduythanh.com>

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý.

Bài 65: Dynamic List

Yêu cầu:

Thiết kế Web Node với nội dung như hình minh họa:

Web Node

THÊM NODE	NỘI DUNG	<input type="text"/>	VỊ TRÍ	<input type="text"/>
XÓA NODE			VỊ TRÍ	<input type="text"/>
SỬA	NỘI DUNG SỬA	<input type="text"/>	VỊ TRÍ	<input type="text"/>

- Data base
- Web1
- Web2

- Thêm Node: Thêm một Node có nội dung vào vị trí trong Web Node (thẻ div chứa các li)
- Xóa Node: Xóa Node tại vị trí nào trong li
- Sửa Node: Thay thế nội dung cũ tại vị trí thành nội dung mới

Hướng dẫn:

Dùng Javascript để xử lý cho các button trong trang Web, mỗi một chức năng viết hàm riêng để xử lý. Sử dụng các hàm insertBefore, appendChild, removeChild, replaceChild...

Bài 66: XML-Dom Parser

Yêu cầu:

Trong file HTML có một chuỗi XML với nội dung sau:

```
<sinhvien>
    <mssv>K123456789</mssv>
    <hoTen>Quách Thị Bán Bún Bò</hoTen>
    <ngaySinh>7/7/1997</ngaySinh>
    <gioiTinh>Nữ</gioiTinh>
</sinhvien>
```

Hãy dùng DOMParser để đọc Mã số sinh viên, Họ tên, Ngày Sinh, Giới tính và hiển thị lên Table:

Chi tiết thông tin Sinh Viên	
Mã Sinh Viên	K123456789
Họ tên	Quách Thị Bán Bún Bò
Ngày Sinh	7/7/1997
Giới Tính	Nữ

- Thêm sự kiện di chuyển chuột vào dòng: Tô nền Xanh dương, chữ trắng
- Thêm sự kiện di chuyển chuột ra khỏi dòng: Tô nền trắng, chữ đen

Hướng dẫn:

Tham khảo cách dùng DOMParser trong slide bài giảng

Bài 67: XML – DOM Parser (*)

Yêu cầu:

Trong file HTML có một chuỗi XML với nội dung sau:

```

<sinhvien>
  <sinhvien>
    <mssv>K123456789</mssv>
    <hoTen>Quách Thị Bán Bún Bò</hoTen>
    <ngaySinh>7/7/1997</ngaySinh>
    <gioiTinh>Nữ</gioiTinh>
  </sinhvien>
  <sinhvien>
    <mssv>K12345000</mssv>
    <hoTen>Hồng Lâu Mộng</hoTen>
    <ngaySinh>2/2/1992</ngaySinh>
    <gioiTinh>Nam</gioiTinh>
  </sinhvien>
  <sinhvien>
    <mssv>K1674000</mssv>
    <hoTen>Hồ Văn Đồ</hoTen>
    <ngaySinh>2/7/1997</ngaySinh>
    <gioiTinh>Nam</gioiTinh>
  </sinhvien>
  ....
</sinhvien>

```

Hãy dùng DOMParser để đọc Mã số sinh viên, Họ tên, Ngày Sinh, Giới tính và hiển thị lên Table:

Danh sách Sinh Viên			
Mã Sinh Viên	Họ Tên	Ngày Sinh	Giới Tính
K123456789	Quách Thị Bán Bún Bò	7/7/1997	Nữ
K12345000	Hồng Lâu Mộng	2/2/1992	Nam
K1674000	Hồ Văn Đồ	2/7/1997	Nam

- Bổ sung sự kiện bấm vào tiêu đề cột: Sắp tăng hoặc giảm theo cột
- Bổ sung sự kiện di chuyển chuột vào cho dòng: tô nền vàng
- Bổ sung sự kiện di chuyển chuột ra khỏi dòng: tô nền trắng
- Bổ sung sự kiện click vào dòng → mở ra một trang chi tiết, ví dụ click vào dòng bạn Bún Bò thì hiển thị trang chi tiết như sau:

Chi tiết thông tin Sinh Viên	
Mã Sinh Viên	K123456789
Họ tên	Quách Thị Bán Bún Bò
Ngày Sinh	7/7/1997
Giới Tính	Nữ

Hướng dẫn:

Tham khảo cách dùng DOMParser trong slide bài giảng

Bài 68: XML - AJAX**Yêu cầu:**

Cho 1 tập tin dữ liệu XML có nội dung:

```
cd_catalog.xml ④ X
1  <?xml version="1.0" encoding="UTF-8"?>
2  <CATALOG>
3   <CD>
4     <TITLE>Empire Burlesque</TITLE>
5     <ARTIST>Bob Dylan</ARTIST>
6     <COUNTRY>USA</COUNTRY>
7     <COMPANY>Columbia</COMPANY>
8     <PRICE>10.98</PRICE>
9     <YEAR>1985</YEAR>
10    </CD>
11   <CD>...</CD>
19   <CD>...</CD>
27   <CD>...</CD>
35   <CD>...</CD>
43   <CD>...</CD>
51   <CD>...</CD>
59   <CD>...</CD>
```

Tải file ở đây:

<https://tranduythanh.com/webmaterials/Bai68.zip>

Dùng AJAX để đọc file cd_catalog.xml hiển thị lên Table như hình:

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr.Hook	Sylvias Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times

Hướng dẫn:

Xem hướng dẫn cách dùng AJAX trong slide bài giảng.

Bài 69: XML – AJAX (*)**Yêu cầu:**

Cho File XML có cấu trúc và dữ liệu như dưới đây:

```
<?xml version="1.0" encoding="UTF-8"?>
<employees>
    <employee id="1" title="Architect">
        <name>Đoàn Ái Nương</name>
        <phone>01656152042</phone>
    </employee>
    <employee id="2" title="Engineer">
        <name>Đoàn Chính Thuần</name>
        <phone>01655654534</phone>
    </employee>
    <employee id="3" title="Teacher">
        <name>Trần Chính Nghĩa</name>
        <phone>0981234567</phone>
    </employee>
    <employee id="4" title="Architect">
        <name>Châu Bá Thông</name>
        <phone>0978976665</phone>
    </employee>
    <employee id="5" title="Engineer">
        <name>Kiều Phong</name>
        <phone>0901334556</phone>
    </employee>
</employees>
```

Sử dụng AJAX để đọc dữ liệu theo yêu cầu:

- Thuộc tính title của các Employee → gom nhóm không trùng nhau và hiển thị lên Dropdown
- Chọn dữ liệu trong Dropdown → Hiển thị Danh sách Employee theo Title trong Dropdown (dùng Table để hiển thị):

Chọn Title:	Architect ▾	
Mã Employee	Tên Employee	Phone Employee
1	Đoàn Ái Nương	01656152042
4	Châu Bá Thông	0978976665

Hướng dẫn:

Xem hướng dẫn cách dùng AJAX trong slide bài giảng, tra Google.

Bài 70: JSONObject vs JSONArray**Yêu cầu:**

Trình bày cấu trúc của JSONObject, JSONArray. So sánh sự khác biệt giữa XML với Json.

Hướng dẫn:

+ Đọc lý thuyết trong Slide và tìm hiểu thêm trên Google

Bài 71: Hiển thị thông tin User**Yêu cầu:**

Cho một đối tượng user có định dạng Json Object như dưới đây (dữ liệu trong file HTML):

```
<script>
var user = {
    "id": "100005823642721",
    "first_name": "Duy Thanh",
    "gender": "male",
    "last_name": "Trần",
    "link": "https://www.facebook.com/duythanhcse",
    "locale": "en_US",
    "birthday": "20/12/1961",
    "name": "Duy Thanh Trần",
    "username": "duythanhcse"
}
</script>
```

Hãy hiển thị thông tin chi tiết của User lên Table.

Hướng dẫn:

+ Xem hướng dẫn trong Slide bài học về JSON

Bài 72:Danh sách sản phẩm**Yêu cầu:**

Cho một JSONArray có cấu trúc và dữ liệu như sau (dữ liệu trong file HTML):

```
"Sanphams": [
    {
        "MaSP": "sp_1xxx",
        "TenSP": "DELL Inspiron 14NR",
        "SoLuong": 100,
        "DonGia": 150000
    },
    {
        "MaSP": "sp_2yyyy",
        "TenSP": "HP Inspiron 113",
        "SoLuong": 130,
        "DonGia": 140000
    },
    ...
]
```

Hãy hiển thị danh sách các Sản phẩm lên Table:

Hiển thị			
Mã	Tên	Số lượng	Đơn giá
sp_1xxx	DELL Inspiron 14NR	100	150000
sp_2yyyy	HP Inspiron 113	130	140000

Bài 73: Danh sách sản phẩm**Yêu cầu:**

Dùng JSONArray để hiển thị dữ liệu sau lên Table (dữ liệu trong file HTML):

```
var sanphams = [
    {
        ma: "sp1",
        ten: "Cocacola",
        gia: 15000
    },
    {
        ma: "sp2",
        ten: "Pepsi",
        gia: 25000
    },
    {
        ma: "sp3",
        ten: "Sting",
        gia: 17000
    },
]
```

The screenshot shows a browser window with a search bar labeled "Muốn tìm gì:" containing a placeholder "Muốn tìm gì:". Below the search bar is a table with the following data:

Mã Sản Phẩm	Tên Sản Phẩm	Đơn giá
sp4	Chanh dây	12000
sp1	Cocacola	15000
sp3	Sting	17000
sp5	pho mát	18000
sp2	Pepsi	25000

Hướng dẫn:

- + Xem hướng dẫn trong Slide bài học về Json

Bài 74: Hiển thị bảng điểm- JSON-AJAX**Yêu cầu:**

Dùng JSon để hiển thị dữ liệu Bảng điểm của Sinh Viên lên Table, yêu cầu dùng AJAX để đọc dữ liệu (dữ liệu nằm ngoài file HTML):

```
▼ array [4]
  ▼ 0 {3}
    Ma : sv1
    Ten : Nguyễn Thị Lan
    ▼ MonHocs [3]
      ▼ 0 {3}
        MaMonHoc : MH1
        TenMH : Toán
        Diem : 9.5
      ▼ 1 {3}
        MaMonHoc : MH2
        TenMH : Lý
        Diem : 7.5
      ▼ 2 {3}
        MaMonHoc : MH3
        TenMH : Hóa
        Diem : 8
    ▶ 1 {3}
    ▶ 2 {3}
  ▶ 3 {3}
```

File dữ liệu databaseSinhVien.json này được lưu trong thư mục Bai74

<https://tranduythanh.com/webmaterials/Bai74.zip>

Muốn tìm gì:						
Mã Sinh Viên	Tên Sinh Viên	Điểm thành phần			Điểm Trung Bình	Kết quả
		Toán	Lý	Hóa		
sv1	Nguyễn Thị Lan	9.5	7.5	8	8.33	Đậu
sv2	Nguyễn Văn Chanh	2.5	8.5	9	6.67	Đậu
sv3	Trần Văn Hò	3.5	2	4.5	3.33	Rớt
sv4	Phạm Ngọc Đồ	4.5	9	2	5.17	Đậu

Hướng dẫn:

- + Xem hướng dẫn trong Slide bài học về Json

Bài 75: Hiển thị Sách môn học - JsonArray**Yêu cầu:**

Cho cấu trúc và dữ liệu của một Json trong HTML như sau:

```
sinhvien={  
    Ma:1,  
    Ten:'Trần Duy Thành',  
    Sachs:  
    [  
        {Ma:'S1',Ten:'Hồng Lâu Mộng',Trang:100},  
        {Ma:'S2',Ten:'Tây Du Ký',Trang:200},  
        {Ma:'S3',Ten:'Tam Quốc Chí',Trang:90},  
        {Ma:'S4',Ten:'Bích Huyết Kiếm',Trang:70},  
        {Ma:'S5',Ten:'Anh Hùng Xạ Đìêu',Trang:1000},  
        {Ma:'S6',Ten:'Thần Đài Đại Hiệp',Trang:500},  
        {Ma:'S7',Ten:'Tần Thủu Hoàng',Trang:600},  
        {Ma:'S8',Ten:'Chiến Quốc',Trang:400},  
        {Ma:'S9',Ten:'Hán Sở Tranh Hùng',Trang:300},  
        {Ma:'S10',Ten:'Bảo Đao',Trang:700},  
    ],  
    ChiTiet:function(){  
        sv=$scope.sinhvien  
        return sv.Ma +" "+sv.Ten  
    }  
}
```

Hãy dùng đọc dữ liệu và hiển thị lên Table như hình dưới đây:

Nhập Mã:	<input type="text"/>	
Nhập Tên:	<input type="text"/>	
Mã và Tên:	<input type="text"/>	
Danh sách Sách đang mượn:		
Mã Sách	Tên Sách	Số Trang
S4	Bích Huyết Kiếm	70
S3	Tam Quốc Chí	90
S1	Hồng Lâu Mộng	100
S2	Tây Du Ký	200
S9	Hán Sở Tranh Hùng	300
S8	Chiến Quốc	400
S6	Thần Đài Đại Hiệp	500
S7	Tần Thủu Hoàng	600
S10	Bảo Đao	700
S5	Anh Hùng Xạ Đìêu	1000

Hướng dẫn:

- + Xem hướng dẫn trong Slide bài học về Json, AngularJS, Google search

Module 7: AngularJS – Binding - Component

Nội dung kiến thức thực hành:

- + Cấu hình và cài đặt NodeJS, AngularJS
- + Thực hành các lệnh của AngularJS: Tạo và chạy dự án, tạo component, service
- + Tìm hiểu cấu trúc thành phần của một Project AngularJS
- + Lập trình Binding: Binding, Binding Property, Binding Style, Binding Event, Binding Two Way
- + Lập trình Type Script
- + Lập trình built-in directives: ngIf, ngSwitch, ngFor
- + Lập trình Component

Bài 76 – Kiến trúc thành phần của Angular

Yêu cầu:

Hãy trình bày kiến trúc thành phần của Angular, giải thích chức năng của từng thành phần (ví dụ App, Module, Component, Service...).

Bài 77 – Cài đặt và lập trình Angular

Yêu cầu:

Hãy trình bày cách thức cài đặt AngularJS

Trình bày các lệnh tạo dự án (đặt tên **my-app**), chạy dự án, tạo component, tạo service

Bài 78 - Binding

Yêu cầu:

Tạo một Component tên “MyComponent”. File Type Script (.ts) chứa một biến tên là “myVar” có giá trị “Hello Angular”, hãy viết các lệnh binding để hiển thị dữ liệu lên giao diện (html) bằng 2 cách:

- Truy suất trực tiếp vào biến “myVar”

- Viết hàm getMyVar() trong (.ts) trả giá trị của “myVar”

File (html) truy suất và hiển thị giá trị của “myVar” trong (.ts) như sau:

- Hiển thị dữ liệu gốc của biến “myVar”
- Hiển thị toàn bộ chữ in hoa của “myVar”
- Hiển thị toàn bộ chữ in thường của “myVar”

Bài 79 – Binding Property

Yêu cầu:

Tạo một “BindingPropertyComponent”, Type Script (.ts) chứa các Property (Attribute/Variable) sau:

```
export class BindingPropertyComponent {  
    public name:string="Trần Duy Thành"  
    public email:string="thanhtd@uel.edu.vn"  
    public nameid:string="nameid"  
    public emailid:string="emailid"  
    public isEnabled:boolean=true  
}
```

HTML sử dụng 2 cách: tạo cấu trúc HTML trong .ts và cấu trúc HTML trong file HTML để binding các thuộc tính này. Sử dụng cú pháp `[]` để binding cho name và `{()}` để binding cho email. Thuộc tính isEnabled sẽ thiết lập cho Email, sinh viên thay đổi true hoặc false để quan sát kết quả.

Bài 80 – Binding Class

Yêu cầu:

Tạo một “BindingClassComponent”.

File “binding-class.component.css” định nghĩa các class sau:

```
.text-success{  
    color:darkgreen  
}  
.text-error{  
    color:darkred  
}  
.text-primary{  
    color:navy  
}
```

```
.text-normal{  
    font-style: italic;  
}
```

File “binding-class.component.ts” khai báo:

```
export class BindingClassComponent {  
    public success="text-success"  
    public error="text-error"  
    public primary="text-primary"  
    public normal="text-normal"  
    public multiClass={  
        "text-primary":true,  
        "text-normal":true,  
        "text-error":true  
    }  
}
```

Yêu cầu, file “binding-class.component.html” tạo 5 thẻ <h1> dùng cú pháp [] để binding các class, tạo 5 thẻ <h2> dùng cú pháp {{}} để binding các class. Nội dung hiển thị trong mỗi <h1> và <h2> là tùy ý.

Sau đó định nghĩa một class “.text-complexity” định dạng chữ Đậm, In Nghiêng, Font chữ “Cambria”, cỡ chữ 18px, màu chữ Blue. Tiến hành định nghĩa tên biến trong .ts và binding trong html để sử dụng class “.text-complexity” này.

Bài 81 – Binding Style

Yêu cầu:

Tạo một “BindingStyleComponent” component.

Định nghĩa Type Script (.ts) như sau:

```
export class BindingStyleComponent {  
    public isError:boolean=false  
    public textStyle={  
        color:'darkorange',  
        fontSize:'26px'  
    }  
}
```

Trong html binding style theo cú pháp sau:

```
<h1 style="color:red;">Trần Duy Thành</h1>  
<h1 [style.color]="isError? 'red':'darkgreen'">Trần Phạm Thành Trà</h1>  
<h1 [style]="{{'color':'purple','font-size':'26px'}}">Trần Phạm Mẫn Nhi</h1>  
<h1 [style]="textStyle">Phạm Thị Xuân Diệu</h1>
```

Mở rộng: Bổ sung style cho chữ tự động in Hoa toàn bộ. Gán binding và kiểm tra kết quả.

Bài 82 – Binding Event

Yêu cầu:

Tạo một “BindingEventComponent” component.

File Type Script “binding-event.component.ts” định nghĩa 2 sự kiện:

```
export class BindingEventComponent {  
    onClick(event:any){  
        alert(event.pointerId)  
    }  
    onSubmit(value:string){  
        alert(value)  
    }  
}
```

File “binding-event.component.html” định nghĩa và binding event:

```
<p>binding-event works!</p>  
<button (click)="onClick($event)">Click Me</button>  
Name:  
<input type="text" #myName>  
<button (click)="onSubmit(myName.value)">Submit</button>
```

Chạy component và quan sát kết quả.

Mở rộng: Chính sửa giao diện cho người dùng nhập 2 số a, b bất kỳ và 5 button:

Button cộng: xử lý xuất kết quả tổng a+b

Button trừ: xử lý xuất kết quả hiệu a-b

Button nhân: xử lý xuất kết quả tích a*b

Button chia: xử lý xuất kết quả thương a/b, lưu ý xử lý mẫu số =0

Button xóa trắng: Xóa dữ liệu trên giao diện.

Chạy component để kiểm tra kết quả sau khi chỉnh sửa.

Bài 83 – Binding Two-Way**Yêu cầu:**

Tạo một component “BindingTwoWayComponent”.

File “binding-two-way.component.ts” định nghĩa:

```
export class BindingTwoWayComponent {  
    public name:string=''  
    public address:string=''  
    setDefaultAddress(){  
        this.address='13 đường Hùng Vương'  
    }  
}
```

File “binding-two-way.component.html” định nghĩa:

```
<p>binding-two-way works!</p>  
<p>Name: <input [(ngModel)]="name" type="text"></p>  
<p>Your name:{{name}}</p>  
<p>Address:<input [(ngModel)]="address" type="text"></p>  
<p>Your address:{{address}}</p>  
<button (click)="setDefaultAddress()">Set default</button>
```

Chạy component và quan sát kết quả.

Mở rộng bổ sung email, số điện thoại, tiến hành binding và chạy lại component, quan sát kết quả sau khi thay đổi.

Bài 84 - Binding Two-Way, model for Quadratic Equation**Yêu cầu:**

Tạo một component có giao diện HTML như sau:

Giải phương trình bậc 2 online	
Hệ số a:	<input type="text" value="1"/>
Hệ số b:	<input type="text" value="3"/>
Hệ số c:	<input type="text" value="-4"/>
Kết quả:	x1=-4 ; x2=1
	<input type="button" value="Giải phương trình"/> <input type="button" value="Xóa trắng"/>

- Dùng Binding Two-way để mapping giá trị cho các biến trong Type Script và Html.

- Tạo một class tên là “Quadratic” nhận vào 3 hệ số a, b,c. và có một phương thức “findSolution()” để xử lý giải phương trình bậc 2
- Khi nhấn nút “Giải phương trình” thì Type Script sẽ áp dụng cơ chế binding để nhận các giá trị nhập vào từ giao diện Html, sau đó dùng lớp “Quadratic” để giải phương trình, sau khi giải xong thì trả về kết quả lên giao diện html.
- Khi nhấn nút “Xóa trắng” thì xóa các dữ liệu trên các ô nhập liệu và dòng kết quả.

Bài 85- Binding Two-Way, model for Lunar Year (*)**Yêu cầu:**

Tạo một component có giao diện như dưới đây:

Chuyển đổi Dương lịch thành Âm lịch						
15	Ngày	5	Tháng	1986	Năm	Chuyển đổi
Âm Lịch						
Thứ trong tuần	Ngày thứ 5					
Ngày tháng năm âm	7/4/1986					
Năm	Bính Dần					
Tháng	Quý Ty					
Ngày	Kỷ Mùi					

Người dùng lựa chọn Ngày tháng năm sinh trên giao diện (dùng select/dropdownlist). Bấm nút “Chuyển đổi”, chuyển ngày dương thành ngày âm.

Yêu cầu dùng cơ chế binding để nạp dữ liệu lên dropdownlist. Viết class “LunarYear” nhận vào 3 thông số (ngày, tháng, năm dương lịch) và hàm “findLunarYearDetail()” trả về thông tin chi tiết như trên giao diện.

Hướng dẫn:

Google search để biết cách chuyển 1 năm Dương lịch thành Âm lịch, cũng như cách binding lấy dữ liệu trên dropdownlist mà người dùng chọn.

Binding dữ liệu từ mảng lên dropdownlist. Ví dụ ta có:

```
days=[ "1", "2", "3"]
<select class="form-select" id="days" name="days">
  <option *ngFor="let day of days">{{day}}</option>
</select>
```

Module 8: AngularJS(tt) – Service - Routing

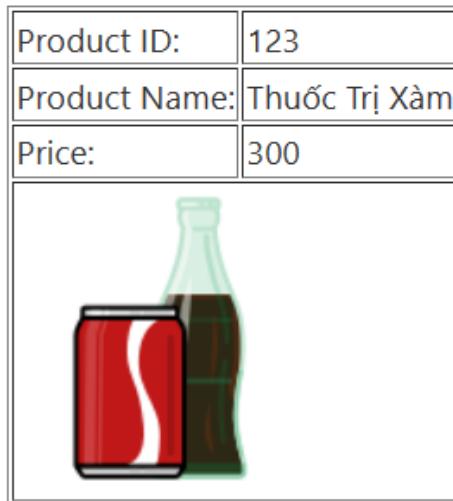
Nội dung kiến thức thực hành:

- + Lập trình built-in directives: ngIf, ngSwitch, ngFor
- + Lập trình Service
- + Lập trình Routing
- + Và các kỹ năng thực hành khác

Bài 86- Json Object Model - Product

Yêu cầu:

Tạo một Component để hiển thị sản phẩm như hình bên dưới. Yêu cầu dùng JsonObject và cơ chế binding. Dữ liệu khai báo trong Type Script.



Bài 87- Json Array Model - Product

Yêu cầu:

Tạo một Component để hiển thị danh sách sản phẩm như hình bên dưới. Yêu cầu dùng JSONArray và cơ chế binding. Dữ liệu khai báo trong Type Script.

Ma San Pham	Ten San Pham	Gia San Pham	Picture
p1	Coca	100	
p2	Pepsi	300	
p3	Sting	200	

Bài 88- Json Array Model – Product Event (*)

Yêu cầu:

Tạo một Component để hiển thị danh sách sản phẩm như hình bên dưới. Yêu cầu dùng JSONArray và cơ chế binding. Dữ liệu khai báo trong Service.

Ma San Pham	Ten San Pham	Gia San Pham	Picture	#
p1	Coca	100		Details
p2	Pepsi	300		Details
p3	Sting	200		Details

Nhấn vào details sẽ hiển thị chi tiết của sản phẩm đó:



Nhấn “Go Back” sẽ quay lại màn hình danh sách sản phẩm.

Hướng dẫn:

- Tạo ProductService:

```
export class ProductService {
  productsImage=[
    {"ProductId":"p1","ProductName":"Coca","Price":100,"Image":"assets/h1.png"},
    {"ProductId":"p2","ProductName":"Pepsi","Price":300,"Image":"assets/h2.png"},
    {"ProductId":"p3","ProductName":"Sting","Price":200,"Image":"assets/h3.png"},
  ]
  constructor() { }
  getProductsWithImages()
  {
    return this.productsImage
  }
  getProductDetail(id:any){

    return this.productsImage.find(x=>x.ProductId==id)
  }
}
```

- Các hình ảnh sao chép vào assets
- Tạo component “service-product-image-event”:

+ Viết lệnh cho “service-product-image-event.component.ts”:

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';
import { ProductService } from '../product.service';

@Component({
  selector: 'app-service-product-image-event',
  templateUrl: './service-product-image-event.component.html',
  styleUrls: ['./service-product-image-event.component.css']
})
export class ServiceProductImageEventComponent {
```

```
public products:any
constructor(pservice: ProductService, private router: Router){
    this.products=pservice.getProductsWithImages()
}
viewDetail(f:any)
{
    this.router.navigate(['service-product-image-event', f.ProductId])
}
```

+ Viết lệnh cho “service-product-image-event.component.html”:

```
<p>service-product-image-event works!</p>
<table border="1">
<tr>
    <td>Ma San Pham</td>
    <td>Ten San Pham</td>
    <td>Gia San Pham</td>
    <td>Picture</td>
    <td>#</td>
</tr>
<tbody *ngFor="let p of products">
    <tr>
        <td>{{p.ProductId}}</td>
        <td>{{p.ProductName}}</td>
        <td>{{p.Price}}</td>
        <td>
            Details</a>
        </td>
    </tr>
</tbody>
</table>
```

- Tạo component “service-product-image-event”:

+ viết lệnh cho “service-product-image-event-detail.component.ts”:

```
import { Component } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { ProductService } from '../product.service';

@Component({
    selector: 'app-service-product-image-event-detail',
    templateUrl: './service-product-image-event-detail.component.html',
    styleUrls: ['./service-product-image-event-detail.component.css']
})
export class ServiceProductImageEventDetailComponent {
    selectedProduct:any
```

```

constructor(private activateRoute: ActivatedRoute, private _fs: ProductService,
private router: Router)
{
  activateRoute.paramMap.subscribe(
    (param)=>{
      let id=param.get('id')

      if(id!=null)
      {
        this.selectedProduct=_fs.getProductDetail(id)
      }
    }
  )
  goBack(){
    this.router.navigate(['service-product-image-event'])
  }
}

```

+ viết lệnh cho “service-product-image-event-detail.component.html”:

```

<p>service-product-image-event-detail works!</p>
<button (click)="goBack()">Go Back</button>
<table>
  <tr>
    <td>ID:</td>
    <td>{{selectedProduct.ProductId}}</td>
  </tr>
  <tr>
    <td>Name:</td>
    <td>{{selectedProduct.ProductName}}</td>
  </tr>
  <tr>
    <td>Price:</td>
    <td>{{selectedProduct.Price}}</td>
  </tr>
  <tr>
    <td colspan="2">
      <img [src]="selectedProduct.Image"/>
    </td>
  </tr>
</table>

```

- Cấu hình Routing:

```

{path: 'service-product-image-event',
component: ServiceProductImageEventComponent},
{path: 'service-product-image-event/:id',
component: ServiceProductImageEventDetailComponent},

```

Bài 89- Json Array Model – Product - Catalog**Yêu cầu:**

Cho CatalogService có cấu trúc như dưới đây:

```
export class CatalogService {  
    datas=[  
        {"Cateid":"cate1","CateName":"nuoc ngọt",  
        "Products": [  
            {"ProductId":"p1","ProductName":"Coca","Price":100,  
            "Image":"assets/h1.png"},  
            {"ProductId":"p2","ProductName":"Pepsi","Price":300,  
            "Image":"assets/h2.png"},  
            {"ProductId":"p3","ProductName":"Sting","Price":200,  
            "Image":"assets/h3.png"},  
        ]},  
        {"Cateid":"cate2","CateName":"Bia",  
        "Products": [  
            {"ProductId":"p4","ProductName":"Heleiken","Price":500,  
            "Image":"assets/h4.png"},  
            {"ProductId":"p5","ProductName":"333","Price":400,  
            "Image":"assets/h5.png"},  
            {"ProductId":"p6","ProductName":"Sai Gon","Price":600,  
            "Image":"assets/h6.png"},  
        ]},  
    ]  
  
    constructor() { }  
    getCategories()  
    {  
        return this.datas  
    }  
}
```

- Tạo component và dùng ngFor lồng nhau để hiển thị danh mục và sản phẩm như hình dưới đây:

Ma Danh Muc		Ten Danh Muc	
cate1		nuoc ngọt	
Ma San Pham	Ten San Pham	Gia San Pham	Picture
p1	Coca	100	
p2	Pepsi	300	
p3	Sting	200	

cate2		Bia	
Ma San Pham	Ten San Pham	Gia San Pham	Picture
p4	Heleiken	500	
p5	333	400	
p6	Sai Gon	600	

Bài 90- Json Array Model – Product– Http Service(*)

Yêu cầu:

Tạo một cơ sở dữ liệu json trong “assets/data/products.json” có cấu trúc mẫu:

```
[{"ProductId": "p1", "ProductName": "Coca", "Price": 100, "Image": "assets/h1.png"}, {"ProductId": "p2", "ProductName": "Pepsi", "Price": 300, "Image": "assets/h2.png"}, {"ProductId": "p3", "ProductName": "Sting", "Price": 200, "Image": "assets/h3.png"}]
```

Hãy dùng cơ chế Service dạng Http để hiển thị dữ liệu như hình:

Ma San Pham	Ten San Pham	Gia San Pham	Picture
p1	Coca	100	
p2	Pepsi	300	
p3	Sting	200	

Hướng dẫn:

-Tạo Interface:

```
export interface IProduct{  
    ProductId:string,  
    ProductName:string,  
    Price:number,  
    Image:string  
}
```

(Có thể để trống toàn bộ các property):

```
export interface IProduct{  
}  
}
```

-Tạo “ProductHttpService”:

```
export class ProductHttpService {  
  
    private _url:string=".//assets//data//products.json";  
    constructor(private _http: HttpClient) { }  
  
    getProducts():Observable<IProduct[]>{  
        return this._http.get<IProduct[]>(this._url)  
    }  
}
```

-Tạo component “service-product-http”:

+ File “service-product-http.component.ts”:

```
export class ServiceProductHttpComponent {  
    products:any;  
    constructor(private _service: ProductHttpService){  
        this._service.getProducts().subscribe({  
            next:(data)=>{this.products=data}  
        })  
    }  
}
```

+ File “service-product-http.component.html”:

```
<p>service-product-http works!</p>  
<table border="1">  
    <tr>  
        <td>Ma San Pham</td>  
        <td>Ten San Pham</td>  
        <td>Gia San Pham</td>
```

```
<td>Picture</td>
</tr>
<tbody>
  <tr *ngFor="let p of products">
    <td>{{p.ProductId}}</td>
    <td>{{p.ProductName}}</td>
    <td>{{p.Price}}</td>
    <td>
      
    </td>
  </tr>
</tbody>
</table>
```

Bài 91- Json Array Model – Product– Http Service Handle Error (*)

Yêu cầu:

Tương tự như bài 90, tuy nhiên bài 91 xử lý lỗi ngoại lệ xảy ra. Ví dụ như cơ sở dữ liệu không tồn tại.

Hướng dẫn:

-Bổ sung “ProductHttpService”, và có tình sửa sai đường dẫn dữ liệu:

```
export class ProductHttpService {

  private _url:string= "./assets/data/productsXXX.json";
  constructor(private _http: HttpClient) { }

  getProducts():Observable<IProduct[]>{
    return this._http.get<IProduct[]>(this._url)
  }
  getProductsHandleError()
  {
    return this._http.get<IProduct[]>(this._url)
      .pipe(retry(3),
        catchError(this.handleError))
  }

  handleError(error:HttpErrorResponse){
    return throwError(()=>new Error(error.message))
  }
}
```

-Tạo component “service-product-http-handle-error”:

+ File “service-product-http-handle-error.component.ts”:

```
export class ServiceProductHttpHandleErrorComponent {  
  products:any  
  errorMessage:string=''  
  constructor(_service:ProductHttpService){  
    _service.getProductsHandleError().subscribe({  
      next:(data)=>{this.products=data},  
      error:(err)=>{this.errorMessage=err}  
    })  
  }  
}
```

+ File “service-product-http-handle-error.component.html”:

```
<p>service-product-http-handle-error works!</p>  
<{{errorMessage}}>  
<table border="1">  
  <tr>  
    <td>Ma San Pham</td>  
    <td>Ten San Pham</td>  
    <td>Gia San Pham</td>  
    <td>Picture</td>  
  </tr>  
  <tbody>  
    <tr *ngFor="let p of products">  
      <td>{{p.ProductId}}</td>  
      <td>{{p.ProductName}}</td>  
      <td>{{p.Price}}</td>  
      <td>  
          
      </td>  
    </tr>  
  </tbody>  
</table>
```

Chạy lên ta có kết quả (trường hợp ta đã sửa sai đường dẫn cơ sở dữ liệu):

service-product-http-handle-error works!

Error: Http failure response for http://localhost:4200/assets/data/products1.json: 404 Not Found

Ma San Pham	Ten San Pham	Gia San Pham	Picture
-------------	--------------	--------------	---------

Nếu sửa lại cho đúng thì ta có kết quả như bài 90.

Bài 92- Json Object Model – Customer – Service**Yêu cầu:**

Tạo một Component để hiển thị Khách hàng như hình bên dưới. Yêu cầu dùng JsonObject và cơ chế binding. Dữ liệu khai báo trong Type Script.

```
customer={  
    "Id":"Cus123",  
    "Name":"Obama",  
    "Email":"obama@gmail.com",  
    "Age":67,  
    "Image":"assets/avatars/obama-avatar.png"  
}
```

Giao diện khi chạy component:



Id:	Cus123
Name:	Obama
Email:	obama@gmail.com
Age:	67

Bài 93- Json Array Model – Group Customers (*)**Yêu cầu:**

Tạo 1 file Json lưu danh sách Khách hàng trong “assets/data/customers.json” có cấu trúc mẫu như dưới đây:

```
[  
    {"CustomerTypeId":1,"CustomerTypeName":"VIP",  
    "Customers":[{
        "Id":"Cus123",
        "Name":"Obama",
        "Email":"obama@gmail.com",
        "Age":67,
        "Image":"assets/avatars/obama-avatar.png"
    }],  
    {
        "Id":"Cus456",
        "Name":"Kim jong Un",
        "Email":"unun@gmail.com",
        "Age":38,
```

```
"Image": "assets/avatars/unun-avatar.png"
}
,
{
  "Id": "Cus789",
  "Name": "Putin",
  "Email": "putin@gmail.com",
  "Age": 77,
  "Image": "assets/avatars/putin-avatar.png"
}]
},
{
  "CustomerTypeId": 2, "CustomerTypeName": "Normal",
  "Customers": [
    {
      "Id": "Cus000",
      "Name": "Hồ Cẩm Đào",
      "Email": "hodao@gmail.com",
      "Age": 16,
      "Image": "assets/avatars/hodao-avatar.png"
    },
    {
      "Id": "Cus111",
      "Name": "Tap Can Binh",
      "Email": "binhbinh@gmail.com",
      "Age": 67,
      "Image": "assets/avatars/binhbinh-avatar.png"
    },
  ]
},
]
```

Áp dụng bài 89 (nhóm) và bài 90 (service http) để làm bài này.

Id	Name	Email	Age	#
1 - VIP				
Cus123	Obama	obama@gmail.com	67	
Cus456	Kim jong Un	unun@gmail.com	38	
Cus789	Putin	putin@gmail.com	77	
2 - Normal				
Cus000	Hồ Cẩm Đào	hodao@gmail.com	16	
Cus111	Tap Can Binh	binhbinh@gmail.com	67	

Bài 94 – Cấu hình Routing

Hiểu đại khái Routing là điều hướng trang, ví dụ ta thấy:

<http://localhost:4200/products>

<http://localhost:4200/customers>

<http://localhost:4200/login>

Khi truy suất vào các link trên và nó điều hướng vào các trang nội dung được định nghĩa thì hiểu là Routing.

Để điều hướng Page ta dùng kỹ thuật Routing, các bước làm như sau:

Bước 1: Tạo các Component, ví dụ ta tạo ProductComponent, ListProductComponent, ServiceProductComponent, ...

Sau đó ta muốn gõ các link sau khi mở các component này:

<http://localhost:4200/product> => Mở ProductComponent

<http://localhost:4200/list-product> => Mở ListProductComponent

<http://localhost:4200/service-product> => Mở ServiceProductComponent

Bước 2: Chính sửa file “app-routing.module.ts”

Khai báo lệnh path và component cho mảng routes như dưới đây:

```
const routes: Routes = [
  {path: 'product', component: ProductComponent},
  {path: 'list-product', component: ListProductComponent},
  {path: 'service-product', component: ServiceProductComponent},
];
```

Cuối file này tiếp tục khai báo biến “RoutingComponent”:

```
export const RoutingComponent=[  
  ProductComponent,  
  ListProductComponent,  
  ServiceProductComponent,  
]
```

Bước 3: Chính sửa file “app.module.ts”:

Thẻ “declarations” bổ sung RoutingComponent mà ta khai báo ở bước 2

```
@NgModule({  
  declarations: [  
    AppComponent,  
    RoutingComponent,  
  ],  
  imports: [  
    BrowserModule,  
    AppRoutingModule,  
    FormsModule,  
    HttpClientModule  
  ],  
  providers: [],  
  bootstrap: [AppComponent]  
})
```

Tới đây chương trình đã hiểu các Routing rồi.

Bước 4: (Bước này không cần thiết)

Giả sử trong ứng dụng (trang chủ) ta muốn điều hướng tới các Routing mà ta đã khai báo thì làm như sau:

Mở file “app.component.html”

```
<table>
  <tr>
    <td>
      <input type="button" value="Product" [routerLink]="'product'">
    </td>
    <td>
      <input type="button" value="List-Product" [routerLink]="'list-product'">
    </td>
    <td>
      <input type="button" value="Service-Product" [routerLink]="'service-product'">
    </td>
  </tr>
</table>
```

Bài 95 – Routing for Page Not Found

Để điều hướng qua Page Not found khi người dùng chọn sai, ta cấu hình dòng cuối trong routes:

```
const routes: Routes = [
  {path: 'product', component: ProductComponent},
  {path: 'list-product', component: ListProductComponent},
  {path: 'service-product', component: ServiceProductComponent},
  {path: "**", component: PageNotFoundComponent},
];
```

Module 9 – AngularJS (tt) - Form

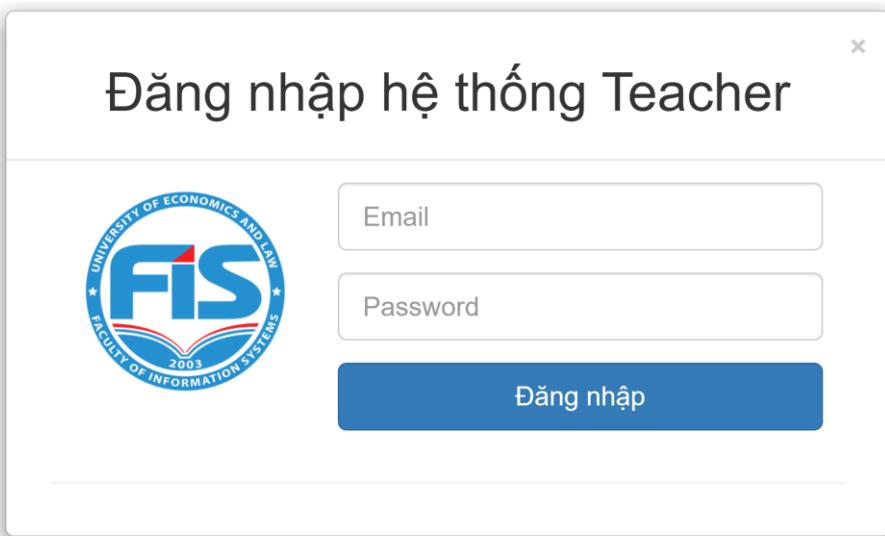
Nội dung kiến thức thực hành:

- + Hiểu được ý nghĩa và chức năng của Form, Reactive Form
- + Sử dụng FormsModule, ngModel
- + Binding Data, Validation
- + Hiểu và sử dụng được các control trên Form: Label, Textbox, Button, Combo, ListBox, TextArea....

Bài 96– Login Screen

Yêu cầu:

Thiết kế giao diện login tương tự như dưới đây:



- Yêu cầu tạo 1 model có thuộc tính email và password
- Khi nhấn nút “Đăng nhập” thì hiển thị JsonObject as string của thông tin và người dùng nhập ngay bên dưới nút “Đăng nhập” (thẻ p, chữ màu đỏ).
- Mở rộng 1: Bổ sung validation: email không hợp lệ, mật khẩu tối thiểu 5 ký tự
- Mở rộng 2: Bổ sung giả lập nếu nhập đúng email và mật khẩu thì navigate tới một component bất kỳ nào đó.
- Mở rộng 3: Bổ sung thêm nút Checkbox(lưu thông tin đăng nhập), tìm từ khóa “Angular Local Storage”. Nếu trước đó checked thì lần sau sẽ nạp lại thông tin đăng nhập khi chạy lại màn hình login.

Bài 97 – Form Đăng ký Khóa học**Yêu cầu:**

Thiết kế màn hình Đăng ký khóa học như hình dưới đây:

Đăng ký khóa học

Tên:

Email:

Điện thoại:

Chọn khóa học:
▼

Chọn ca học: Sáng Tối

Đồng ý với điều khoản

Submit

```
{ "name": "Nam Anh", "email": "anh@gmail.com", "phone": "0909090909", "course": "Ruby", "time": "toi", "agree": true }
```

- Yêu cầu khai báo lớp model để binding dữ liệu
- Khi bấm Submit, thông tin JsonObject as string sẽ hiển thị ngay bên dưới nút Submit.

Hướng dẫn:

- Sử dụng ReactiveFormsModule

Bài 98 – Form Đăng ký Khóa học - Validation**Yêu cầu:**

Mở rộng bài tập Form Đăng ký khóa học, bổ sung Validation

Đăng ký khóa học

Tên:

!

Bạn chưa nhập tên

Email:

Điện thoại:

Chọn khóa học:

!

Vui lòng chọn khóa học

Chọn ca học: Sáng Tối

Đồng ý với điều khoản

Submit

Khi nhấn nút “Submit”:

- Nếu tên rỗng sẽ báo lỗi
- Nếu khóa học chưa chọn sẽ báo lỗi
- Nếu chưa bấm vào “đồng ý với điều khoản” sẽ báo lỗi

Hướng dẫn:

Bài 99- Mathematics

Yêu cầu:

Tạo trang web với Table và Textbox, Button để tính MAX, MIN, sin, cos... Theo mô tả dưới đây:

a(So Thu 1):	30
b(So Thu 2)	45
c(So Thu 3)	90
KET QUA	-0.9880316240928618
MAX	
MIN	
sin(a)	
Cos(a)	
a^b	

- Báo lỗi (tô nền đỏ) các ô nhập liệu nếu sai định dạng dữ liệu số.

Module 10 – AngularJS (tt) Triệu gọi External Restful API

Nội dung kiến thức thực hành:

- + Luyện tập CSS
- + Nhúng Bootstrap
- + Luyện tập Service
- + Luyện tập Component
- + Luyện tập HTTP Call external restful API
- + Luyện tập xử lý Http Handling Error
- + Xử lý CORS Policy

Bài 100: Tương tác External Restful API- Ngân hàng Đông Á

Yêu cầu:

Ngân hàng Đông Á cung cấp Restful API sau để truy suất tỉ giá hối đoái của một số quốc gia:

<https://www.dongabank.com.vn/exchange/export>

Dưới đây là dữ liệu của API trả về:

```
({"items": [{"type": "GBP", "imageurl": "https://www.dongabank.com.vn/images/flag/GBP.gif", "muatienmat": "28050", "muack": "28170", "bantienmat": "28760", "banck": "28710"}, {"type": "AUD", "imageurl": "https://www.dongabank.com.vn/images/flag/AUD.gif", "muatienmat": "15800", "muack": "15890", "bantienmat": "16230", "banck": "16190"}, {"type": "CAD", "imageurl": "https://www.dongabank.com.vn/images/flag/CAD.gif", "muatienmat": "17220", "muack": "17320", "bantienmat": "17690", "banck": "17650"}, {"type": "CHF", "imageurl": "https://www.dongabank.com.vn/images/flag/CHF.gif", "muatienmat": "22610", "muack": "25000", "bantienmat": "23080", "banck": "25500"}, {"type": "CNY", "imageurl": "https://www.dongabank.com.vn/images/flag/CNY.gif", "muatienmat": "3000", "muack": "3000", "bantienmat": "3500", "banck": "3500"}, {"type": "EUR", "imageurl": "https://www.dongabank.com.vn/images/flag/EUR.gif", "muatienmat": "24860", "muack": "24970", "bantienmat": "25490", "banck": "25440"}, {"type": "HKD", "imageurl": "https://www.dongabank.com.vn/images/flag/HKD.gif", "muatienmat": "2410", "muack": "2900", "bantienmat": "2920", "banck": "3060"}, {"type": "XAU", "imageurl": "https://www.dongabank.com.vn/images/flag/XAU.gif", "muatienmat": "6620000", "muack": "6655000", "bantienmat": "6690000", "banck": "6655000"}, {"type": "USD", "imageurl": "https://www.dongabank.com.vn/images/flag/USD.gif", "muatienmat": "23590", "muack": "23590", "bantienmat": "23920", "banck": "23870"}, {"type": "THB", "imageurl": "https://www.dongabank.com.vn/images/flag/THB.gif", "muatienmat": "610", "muack": "670", "bantienmat": "700", "banck": "690"}, {"type": "SGD", "imageurl": "https://www.dongabank.com.vn/images/flag/SGD.gif", "muatienmat": "17310", "muack": "17470", "bantienmat": "17773"}])
```

```
"17820", "banck": "17800"}, {"type": "PNJ_DAB", "imageurl": "https://www.dongabank.com.vn/images/flag/PNJ_DAB.gif", "muatienmat": "5330000", "muack": "5330000", "bantienmat": "5430000", "banck": "5430000"}, {"type": "NZD", "imageurl": "https://www.dongabank.com.vn/images/flag/NZD.gif", "muatienmat": "", "muack": "14650", "bantienmat": "", "banck": "15060"}, {"type": "JPY", "imageurl": "https://www.dongabank.com.vn/images/flag/JPY.gif", "muatienmat": "169", "muack": "172.4", "bantienmat": "176.1", "banck": "175.6"}])}
```

Yêu cầu Sinh viên sử dụng HTTP và handling error để triệu gọi API này và hiển thị lên giao diện như hình dưới đây:

Dong A Bank					
Type	ImageURL	MuaTienMat	MuaCK	BanTienMat	BanCK
CAD		17220	17320	17690	17650
XAU		6620000	6655000	6690000	6655000
AUD		15800	15890	16230	16190
CHF		22610	25000	23080	25500
CNY		3000	3000	3500	3500
EUR		24860	24970	25490	25440
GBP		28050	28170	28760	28710
HKD		2410	2900	2920	3060
JPY		169	172.4	176.1	175.6
NZD			14650		15060
PNJ_DAB		5330000	5330000	5430000	5430000
SGD		17310	17470	17820	17800
THB		610	670	700	690
USD		23590	23590	23920	23870

Hướng dẫn chi tiết:

Như đã được học, dữ liệu Json có 2 loại: JsonObject {} và JSONArray []. Tuy nhiên dữ liệu của Ngân hàng đồng á gửi về lại được bao bọc bởi ngoặc tròn (). Như vậy Ta cần tải dữ liệu và xóa ngoặc () đi để nó trở thành JSONObject (được bao bọc bởi ngoặc nhọn {}).

Quan sát thấy “items” chứa mảng các JSONObject:
 {"type", "imageurl", "muatienmat", "muack", "bantienmat", "banck"}

Như vậy ta cần tạo 2 interface (hoặc 2 class) để mô hình hóa và mapping dữ liệu.

Bước 1: Tạo các lớp mô hình hóa cho dữ liệu

Tạo file “DongABankItem.ts” và khai báo interface như dưới đây:

```
export interface IDongABankItem
{
    type:string,
    imageurl:string,
    muatienmat:string,
    muack:string,
    bantienmat:string,
    banck:string,
}
```

Tiếp tục tạo file “” và khai báo interface như dưới đây:

```
import { IDongABankItem } from "./DongABankItem";

export interface IDongABankData
{
    items:Array<IDongABankItem>
}
```

Như vậy với cách khai báo 2 interface ở trên thì chương trình sẽ mapping được dữ liệu Json mà Ngân Hàng Đông Á gửi về thành mô hình hướng đối tượng.

Điều quan trọng nhất là chúng ta phải hiểu đúng cấu trúc dữ liệu Json mà bất kỳ 1 API nào đó trả về, việc hiểu đúng cấu trúc Json thì ta mới khai báo các mô hình lớp chính xác được.

Các thuộc tính khai báo trong interface hay trong class Ta nên copy + paste từ dữ liệu mà API trả về nhằm tránh bị sai chính tả (phải đúng cả chữ Hoa, chữ Thường).

Bước 2: Khai báo Service

Tạo Service tên “DongABankService” lúc này file “dong-abank.service.ts” sẽ được tạo ra, ta tiến hành coding cho file này.

```
import { JsonPipe } from '@angular/common';
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { catchError, map, Observable, retry, throwError } from 'rxjs';
import { IDongABankData } from './interfaces/DongABankData';
import { IDongABankItem } from './interfaces/DongABankItem';

@Injectable({
    providedIn: 'root'
})
export class DongABankService {

    private _url:string="/exchange/export"
    constructor(private _http: HttpClient) { }
```

```
getDongABankData()
{
    const headers=new HttpHeaders().set("Content-Type", "text/plain; charset=utf-8")
    const requestOptions:Object={
        headers:headers,
        responseType:"text"
    }
    return this._http.get<any>(this._url,requestOptions).pipe(
        map(res=>JSON.parse(res.slice(1,-1)) as IDongABankData),
        retry(3),
        catchError(this.handleError)
    )
}

handleError(error:HttpErrorResponse){
    return throwError(()=>new Error(error.message))
}
}
```

res.slice(1,-1) xóa các ngoặc tròn ở đầu và ở cuối dữ liệu mà API trả về
JSON.parse(res.slice(1,-1)) đưa dữ liệu về JSON

JSON.parse(res.slice(1,-1)) as IDongABankData mô hình hóa Json thành mô hình hướng đối tượng.

handleError giúp kiểm tra lỗi trong quá trình triệu gọi API

Bước 3: Tạo Component

Tạo “DongABankComponent” component, lúc này ta coding trong file “dong-abank.component.ts” như sau:

```
import { Component } from '@angular/core';
import { DongABankService } from '../dong-abank.service';

@Component({
    selector: 'app-dong-abank',
    templateUrl: './dong-abank.component.html',
    styleUrls: ['./dong-abank.component.css']
})
export class DongABankComponent {
    data:any
    errMessage:string=''
    constructor(_service:DongABankService){
        _service.getDongABankData().subscribe({
            next:(data)=>{
                this.data=data
            },
            error:(err)=>{
                this.errMessage=err
            }
        })
    }
}
```

Tiếp tục coding html trong file “dong-abank.component.html”:

```
<h3 class="text-center text-white bg-dark p-3">Dong A Bank</h3>
{{errMessage}}
<div class="container">
  <table class="table table -hover">
    <thead>
      <tr>
        <th>Type</th>
        <th>ImageURL</th>
        <th>MuatienMat</th>
        <th>Muack</th>
        <th>Bantienmat</th>
        <th>BanCK</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let item of data.items">
        <td>{{item.type}}</td>
        <td></td>
        <td>{{item.muatienmat}}</td>
        <td>{{item.muack}}</td>
        <td>{{item.bantienmat}}</td>
        <td>{{item.banck}}</td>
      </tr>
    </tbody>
  </table>
</div>
```

Bước 4: Xử lý CORS Policy

Lưu ý rằng, khi gọi external restful API thông thường ta sẽ gặp lỗi tương tự:

```
✖ Access to XMLHttpRequest at 'https://www.dongabank.com.vn/e localhost/:1
xchange/export' from origin 'http://localhost:4200' has been blocked by
CORS policy: No 'Access-Control-Allow-Origin' header is present on the
requested resource.
```

Do đó ta cần xử lý Xử lý CORS Policy như sau:

- Tạo file src/proxy.conf.json và khai báo cú pháp:

Vì thông thường một dự án có thể kết nối nhiều API khác nhau (tức là mảng) nên ta khai báo nội dung của file proxy này dạng mảng cho tiện lợi:

```
[{
  {
    "context": ["/exchange"],
    "target": "https://www.dongabank.com.vn",
    "secure": true,
    "changeOrigin":true,
    "logLevel": "debug"
  }
}]
```

“context” trả tới API mà ta muốn truy suất

“target” là domain mà API được cấu hình

Các thông số khác thiết lập như minh họa ở trên. Khi triệu gọi nhiều API khác thì cứ Copy + paste cấu trúc {} thành nhiều cái, rồi chỉnh lại context và target là được.

Trong trường hợp nếu dự án chỉ truy suất tới 1 API duy nhất thì ta có thể khai báo dạng Object như dưới đây cho “proxy.conf.json” (nhưng chắc là hiếm, vì 1 dự án thường truy suất tới nhiều API):

```
{
  "/exchange": {
    "target": "https://www.dongabank.com.vn",
    "secure": true,
    "changeOrigin": true,
    "logLevel": "debug"
  }
}
```

Sau khi khai báo xong file “proxy.conf.json”, ta tiến hành cấu hình “angular.json”:

```

{
  "projects": {
    "my-app": {
      "architect": {
        "serve": {
          "builder": "@angular-devkit/build-angular:dev-server",
          "options": {
            "browserTarget": "my-app:build",
            "proxyConfig": "src/proxy.conf.json"
          },
          "configurations": {
            "production": {
              "browserTarget": "my-app:build:production"
            },
            "development": {
              "browserTarget": "my-app:build:development"
            }
          },
          "defaultConfiguration": "development"
        }
      }
    }
  }
}

```

Bổ sung khung màu đỏ.

```

"serve": {
  "builder": "@angular-devkit/build-angular:dev-server",
  "options": {
    "browserTarget": "my-app:build",
    "proxyConfig": "src/proxy.conf.json"
  },
  "configurations": {
    "production": {
      "browserTarget": "my-app:build:production"
    }
  }
}

```

```
        "browserTarget": "my-app:build:production"
    },
    "development": {
        "browserTarget": "my-app:build:development"
    }
},
"defaultConfiguration": "development"
},
```

"browserTarget" trả tới tên ứng dụng mà mình tạo, ví dụ “my-app:build”

"proxyConfig" trả tới file proxy mà mình định nghĩa, ví dụ “src/proxy.conf.json”

Bước 5: Cài đặt BootStrap – Jquery (không cần thiết nếu không dùng)

Thực hiện lệnh:

```
npm install --save bootstrap jquery
```

Tiếp tục cấu hình “angular.json”:

```
12     "architect": {
13         "build": {
14             "builder": "@angular-devkit/build-angular:browser",
15             "options": {
16                 "outputPath": "dist/my-app",
17                 "index": "src/index.html",
18                 "main": "src/main.ts",
19                 "polyfills": [
20                     "zone.js"
21                 ],
22                 "tsConfig": "tsconfig.app.json",
23                 "assets": [
24                     "src/favicon.ico",
25                     "src/assets"
26                 ],
27                 "styles": [
28                     "src/styles.css",
29                     "node_modules/bootstrap/dist/css/bootstrap.min.css"
30                 ],
31                 "scripts": [
32                     "node_modules/jquery/dist/jquery.min.js",
33                     "node_modules/bootstrap/dist/js/bootstrap.min.js"
34                 ]
35             },
36         }
37     }
```

Bổ sung khung màu đỏ.

```
"styles": [
    "src/styles.css",
    "node_modules/bootstrap/dist/css/bootstrap.min.css"
],
"scripts": [
    "node_modules/jquery/dist/jquery.min.js",
    "node_modules/bootstrap/dist/js/bootstrap.min.js"
]
```

Chạy chương trình và ta có được kết quả như mong muốn.

Bài 101: Tương tác External Restful API- Product Service 1

Yêu cầu:

Sinh viên thực hiện tương tự như bài tập triệu gọi API của ngân hàng Đông Á, load danh sách sản phẩm từ API: <https://fakestoreapi.com/products> hiển thị giao diện như sau:

Fake Product Data						
Id	title	price	description	category	image	rating
1	Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops	109.95	Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday	men's clothing		3.9
2	Mens Casual Premium Slim Fit T-Shirts	22.3	Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.	men's clothing		4.1
3	Mens Cotton Jacket	55.99	great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in this thanksgiving or Christmas Day.	men's clothing		4.7
4	Mens Casual Slim Fit	15.99	The color could be slightly different between on the screen and in practice. / Please note that body builds vary by person, therefore, detailed size information should be reviewed below on the product	men's clothing		2.1

Hướng dẫn chi tiết:

Quan sát dữ liệu trả về của API <https://fakestoreapi.com/products>:

```
[{"id":1,"title":"Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops","price":109.95,"description":"Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday","category":"men's clothing","image":"https://fakestoreapi.com/img/81fPKd-2AYL.AC_SL1500.jpg","rating":{"rate":3.9,"count":120}}, {"id":2,"title":"Mens Casual Premium Slim Fit T-Shirts ","price":22.3,"description":"Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.","category":"men's clothing","image":"https://fakestoreapi.com/img/71-3HjGNDUL.AC_SY879.SX.UX.SY.UY.jpg","rating":{"rate":4.1,"count":259}}, {"id":3,"title":"Mens Cotton Jacket","price":55.99,"description":"great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in this thanksgiving or Christmas Day.","category":"men's clothing","image":"https://fakestoreapi.com/img/71li-ujtlUL.AC_UX679.jpg","rating":{"rate":4.7,"count":500}}, {"id":4,"title":"Mens Casual Slim Fit","price":15.99,"description":"The color could be slightly different between on the screen and in practice. / Please note that body builds vary by person, therefore, detailed size information should be reviewed below on the product description.","category":"men's clothing","image":"https://fakestoreapi.com/img/71YXzeOusIL.AC_UY879.jpg","rating":{"rate":2.1,"count":430}}, {"id":5,"title":"John Hardy Women's Legends Naga Gold & Silver Dragon Station Chain Bracelet","price":695,"description":"From our Legends Collection, the Naga was inspired by the"}]
```

mythical water dragon that protects the ocean's pearl. Wear facing inward to be bestowed with love and abundance, or outward for protection.", "category": "jewelery", "image": "https://fakestoreapi.com/img/71pWzhdJNwL.AC_UL640_QL65_ML3.jpg", "rating": {"rate": 4.6, "count": 400}, {"id": 6, "title": "Solid Gold Petite Micropave", "price": 168, "description": "Satisfaction Guaranteed. Return or exchange any order within 30 days. Designed and sold by Hafeez Center in the United States. Satisfaction Guaranteed. Return or exchange any order within 30 days.", "category": "jewelery", "image": "https://fakestoreapi.com/img/61sbMiUnoGL.AC_UL640_QL65_ML3.jpg", "rating": {"rate": 3.9, "count": 70}}, {"id": 7, "title": "White Gold Plated Princess", "price": 9.99, "description": "Classic Created Wedding Engagement Solitaire Diamond Promise Ring for Her. Gifts to spoil your love more for Engagement, Wedding, Anniversary, Valentine's Day...", "category": "jewelery", "image": "https://fakestoreapi.com/img/71YAIFU48IL.AC_UL640_QL65_ML3.jpg", "rating": {"rate": 3, "count": 400}}, {"id": 8, "title": "Pierced Owl Rose Gold Plated Stainless Steel Double", "price": 10.99, "description": "Rose Gold Plated Double Flared Tunnel Plug Earrings. Made of 316L Stainless Steel", "category": "jewelery", "image": "https://fakestoreapi.com/img/51UDEzMJVpL.AC_UL640_QL65_ML3.jpg", "rating": {"rate": 1.9, "count": 100}}, {"id": 9, "title": "WD 2TB Elements Portable External Hard Drive - USB 3.0", "price": 64, "description": "USB 3.0 and USB 2.0 Compatibility Fast data transfers Improve PC Performance High Capacity; Compatibility Formatted NTFS for Windows 10, Windows 8.1, Windows 7; Reformatting may be required for other operating systems; Compatibility may vary depending on user's hardware configuration and operating system", "category": "electronics", "image": "https://fakestoreapi.com/img/61IBBVJvSDL.AC_SY879_.jpg", "rating": {"rate": 3.3, "count": 203}}, {"id": 10, "title": "SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s", "price": 109, "description": "Easy upgrade for faster boot up, shutdown, application load and response (As compared to 5400 RPM SATA 2.5" hard drive; Based on published specifications and internal benchmarking tests using PCMark vantage scores) Boosts burst write performance, making it ideal for typical PC workloads The perfect balance of performance and reliability Read/write speeds of up to 535MB/s/450MB/s (Based on internal testing; Performance may vary depending upon drive capacity, host device, OS and application.)", "category": "electronics", "image": "https://fakestoreapi.com/img/61U7T1koQgL.AC_SX679_.jpg", "rating": {"rate": 2.9, "count": 470}}, {"id": 11, "title": "Silicon Power 256GB SSD 3D NAND A55 SLC Cache Performance Boost SATA III 2.5", "price": 109, "description": "3D NAND flash are applied to deliver high transfer speeds Remarkable transfer speeds that enable faster bootup and improved overall system performance. The advanced SLC Cache Technology allows performance boost and longer lifespan 7mm slim design suitable for Ultrabooks and Ultra-slim notebooks. Supports TRIM command, Garbage Collection technology, RAID, and ECC (Error Checking & Correction) to provide the optimized performance and enhanced reliability.", "category": "electronics", "image": "https://fakestoreapi.com/img/71kWymZ+cL.AC_SX679_.jpg", "rating": {"rate": 4.8, "count": 319}}, {"id": 12, "title": "WD 4TB Gaming Drive Works with Playstation 4 Portable External Hard Drive", "price": 114, "description": "Expand your PS4 gaming experience, Play anywhere Fast and easy, setup Sleek design with high capacity, 3-year manufacturer's limited warranty", "category": "electronics", "image": "https://fakestoreapi.com/img/61mtL65D4cL.AC_SX679_.jpg", "rating": {"rate": 4.8, "count": 400}}, {"id": 13, "title": "Acer SB220Q bi 21.5 inches Full HD (1920 x 1080) IPS Ultra-Thin", "price": 599, "description": "21.5 inches Full HD (1920 x 1080) widescreen IPS display And Radeon free Sync technology. No compatibility for VESA Mount Refresh Rate: 75Hz - Using HDMI port Zero-frame design | ultra-thin | 4ms response time | IPS panel Aspect ratio - 16:9. Color Supported - 16.7 million colors. Brightness - 250 nit Tilt angle -5 degree to 15 degree. Horizontal viewing angle-178 degree. Vertical viewing angle-178 degree 75 hertz", "category": "electronics", "image": "https://fakestoreapi.com/img/81QpkIctqPL.AC_SX679_.jpg", "rating": {"rate": 2.9, "count": 250}}, {"id": 14, "title": "Samsung 49-Inch CHG90 144Hz Curved Gaming Monitor (LC49HG90DMNXZA) - Super Ultrawide Screen QLED", "price": 999.99, "description": "49 INCH SUPER ULTRAWIDE 32:9 CURVED GAMING MONITOR with dual 27 inch screen side by side QUANTUM DOT (QLED) TECHNOLOGY, HDR support and factory calibration provides stunningly realistic and accurate color and contrast 144HZ HIGH REFRESH RATE and 1ms ultra fast response time work to eliminate motion blur, ghosting, and reduce input lag", "category": "electronics", "image": "https://fakestoreapi.com/img/81Zt42ioCgL.AC_SX679_.jpg", "rating": {"rate": 2.2, "count": 140}}, {"id": 15, "title": "BIYLACLESEN Women's 3-in-1 Snowboard Jacket Winter Coats", "price": 56.99, "description": "Note:The Jackets is US standard size, Please choose size as your usual wear Material: 100% Polyester; Detachable Liner Fabric: Warm Fleece. Detachable Functional Liner: Skin Friendly, Lightweight and Warm. Stand Collar Liner jacket, keep you warm in cold weather. Zippered Pockets: 2 Zippered Hand Pockets, 2 Zippered Pockets on Chest (enough to keep cards or keys) and 1 Hidden Pocket Inside. Zippered Hand Pockets and Hidden Pocket keep your things secure. Humanized Design: Adjustable and Detachable Hood and Adjustable cuff to prevent the wind and water, for a comfortable fit. 3 in 1 Detachable Design provide more convenience, you can separate the coat and inner as needed, or wear it together. It is suitable for different season and help you adapt to different climates", "category": "women's clothing", "image": "https://fakestoreapi.com/img/51Y5NI-I5jL.AC_UX679_.jpg", "rating": {"rate": 2.6, "count": 235}}, {"id": 16, "title": "Lock and Love Women's Removable Hooded Faux Leather Moto Biker Jacket", "price": 29.95, "description": "100% POLYURETHANE(shell) 100% POLYESTER(lining) 75% POLYESTER 25% COTTON (SWEATER), Faux leather material for style and comfort / 2 pockets of front, 2-For-One Hooded denim style faux leather jacket, Button detail on waist / Detail stitching at sides, HAND WASH ONLY / DO NOT BLEACH / LINE DRY / DO NOT IRON", "category": "women's clothing", "image": "https://fakestoreapi.com/img/81XH0e8fefL.AC_UY879_.jpg", "rating": {"rate": 2.9, "count": 340}}, {"id": 17, "title": "Rain Jacket Women Windbreaker Striped Climbing Raincoats", "price": 39.99, "description": "Lightweight perfect for trip or casual wear---Long sleeve with hooded, adjustable drawstring waist design. Button and zipper front closure raincoat, fully stripes Lined and The Raincoat has 2 side pockets are a good size to hold all kinds of things, it covers the hips, and the hood is generous but doesn't overdo it. Attached Cotton Lined Hood with"}>

```
Adjustable Drawstrings give it a real styled look.", "category": "women's clothing", "image": "https://fakestoreapi.com/img/71Hb1AHs5xL.AC_UY879_2.jpg", "rating": {"rate": 3.8, "count": 679}, {"id": 18, "title": "MBJ Women's Solid Short Sleeve Boat Neck V ", "price": 9.85, "description": "95% RAYON 5% SPANDEX, Made in USA or Imported, Do Not Bleach, Lightweight fabric with great stretch for comfort, Ribbed on sleeves and neckline / Double stitching on bottom hem", "category": "women's clothing", "image": "https://fakestoreapi.com/img/71z3kpMAYsL.AC_UY879_.jpg", "rating": {"rate": 4.7, "count": 130}, {"id": 19, "title": "Opna Women's Short Sleeve Moisture", "price": 7.95, "description": "100% Polyester, Machine wash, 100% cationic polyester interlock, Machine Wash & Pre Shrunk for a Great Fit, Lightweight, roomy and highly breathable with moisture wicking fabric which helps to keep moisture away, Soft Lightweight Fabric with comfortable V-neck collar and a slimmer fit, delivers a sleek, more feminine silhouette and Added Comfort", "category": "women's clothing", "image": "https://fakestoreapi.com/img/51eg55uWmdL.AC_UX679_.jpg", "rating": {"rate": 4.5, "count": 146}, {"id": 20, "title": "DANVOUY Womens T Shirt Casual Cotton Short", "price": 12.99, "description": "95%Cotton,5%Spandex, Features: Casual, Short Sleeve, Letter Print,V-Neck,Fashion Tees, The fabric is soft and has some stretch., Occasion: Casual/Office/Beach/School/Home/Street. Season: Spring,Summer,Autumn,Winter.", "category": "women's clothing", "image": "https://fakestoreapi.com/img/61pHAEJ4NML.AC_UX679_.jpg", "rating": {"rate": 3.6, "count": 145}}]
```

Nhìn cấu trúc trên thì nó đơn giản hơn dữ liệu của Ngân Hàng Đông Á. Nó là mảng các đối tượng và ngay từ đầu nó đã đúng với cấu trúc của Json Data.

Bước 1: Tạo các interface/lớp mô hình

Đối với "rating": {"rate": 3.6, "count": 145} cần có một interface, đặt tên file "FakeProductRating.ts" và khai báo interface "IFakeProductRating" có nội dung:

```
export interface IFakeProductRating{  
    rate:number,  
    count:number  
}
```

Đối với cấu trúc ở ngoài, tạo file "FakeProduct.ts" và khai báo interface "IFakeProduct" có nội dung:

```
import {IFakeProductRating} from './FakeProductRating';  
  
export interface IFakeProduct{  
    id:number,  
    title:string,  
    price:number,  
    description:string,  
    category:string,  
    image:string ,  
    rate:IFakeProductRating  
}
```

Bước 2: Tạo "FakeProductService" Service, viết mã lệnh trong file "fake-product.service.ts" như sau:

```
import { HttpClient, HttpErrorResponse, HttpHeaders } from  
'@angular/common/http';  
import { Injectable } from '@angular/core';
```

```
import { catchError, map, Observable, retry, throwError } from 'rxjs';
import { IFakeProduct } from './interfaces/FakeProduct';
@Injectable({
  providedIn: 'root'
})
export class FakeProductService {
  private _url:string="/products"
  constructor(private _http: HttpClient) { }
  getFakeProductData():Observable<any> {
    const headers=new HttpHeaders().set("Content-Type","text/plain;charset=utf-8")
    const requestOptions:Object={
      headers:headers,
      responseType:"text"
    }
    return this._http.get<any>(this._url,requestOptions).pipe(
      map(res=>JSON.parse(res) as Array<IFakeProduct>),
      retry(3),
      catchError(this.handleError)
    )
  }
  handleError(error:HttpErrorResponse){
    return throwError(()=>new Error(error.message))
  }
}
```

Bước 3: Tạo Component

Tạo “FakeProductComponent” component, lúc này ta coding trong file “fake-product.component.ts” như sau:

```
import { Component, OnInit } from '@angular/core';
import { FakeProductService } from '../fake-product.service';
@Component({
  selector: 'app-fake-product',
  templateUrl: './fake-product.component.html',
  styleUrls: ['./fake-product.component.css']
})
export class FakeProductComponent {
  data:any
  errMessage:string=''
  constructor(_service:FakeProductService){
    _service.getFakeProductData().subscribe({
      next:(data)=>{ this.data=data},
      error:(err)=>{
        this.errMessage=err
      }
    })
  }
}
```

Chỉnh sửa HTML trong file “fake-product.component.html”:

```
<p>binding-style works!</p>
<table>
  <thead>
    <tr>
      <th>Id</th>
      <th>title</th>
      <th>price</th>
      <th>description</th>
      <th>category</th>
      <th>image</th>
      <th>rating</th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let item of data">
      <td>{{item.id}}</td>
      <td>{{item.title}}</td>
      <td>{{item.price}}</td>
      <td>{{item.description}}</td>
      <td>{{item.category}}</td>
      <td></td>
      <td>{{item.rating.rate}}</td>
    </tr>
  </tbody>
</table>
```

Bước 4: Xử lý CORS Policy

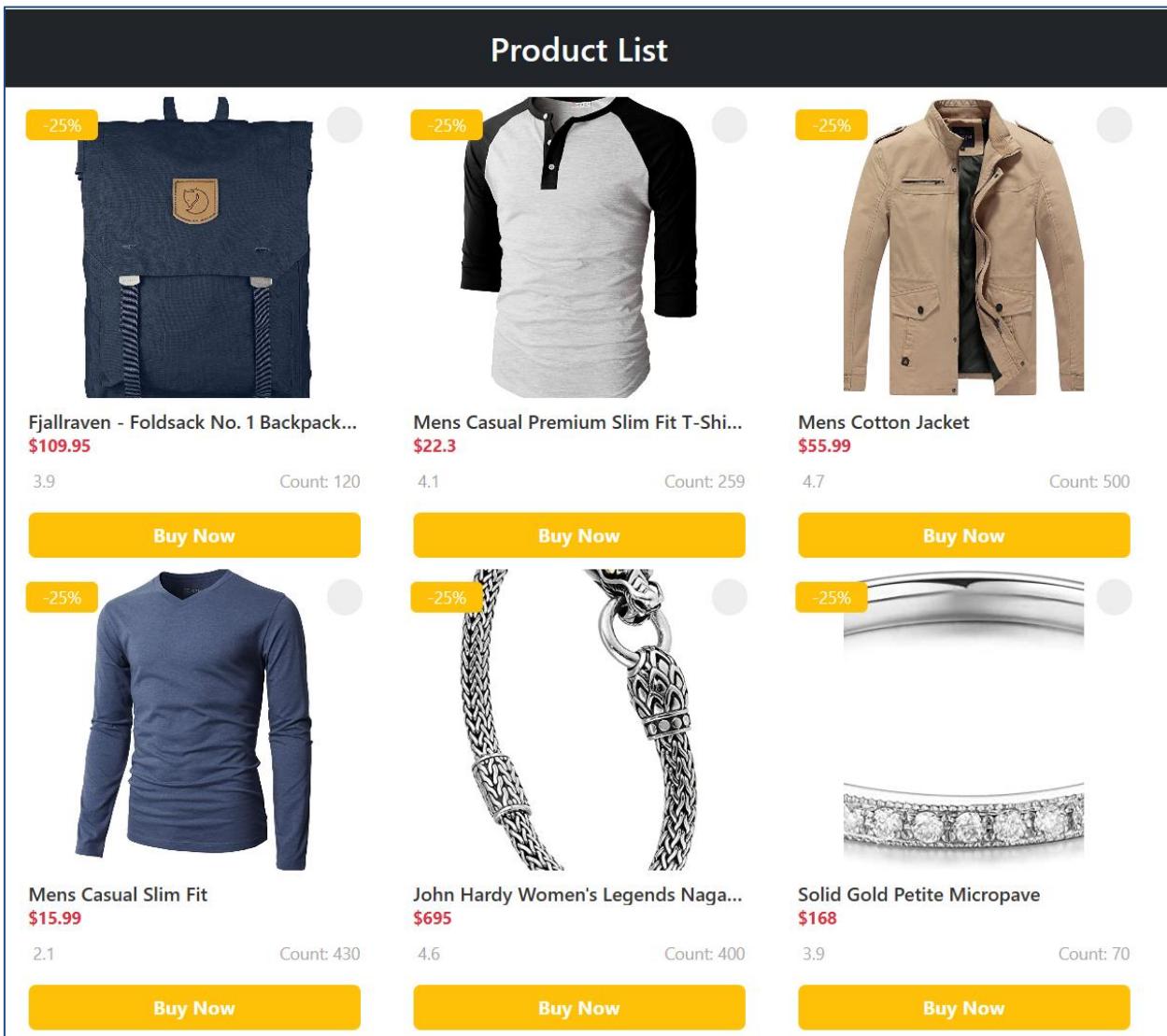
Khai báo và cấu hình tương tự như bài triệu gọi API của Ngân Hàng Đông Á. Ta chỉnh sửa (hoặc tạo mới) “proxy.conf.json”. Như đã giải thích ở bài trước, vì trong 1 dự án có thể kết nối nhiều API nên nội dung file này như sau:

```
[{
  {
    "context": ["/exchange"],
    "target": "https://www.dongabank.com.vn",
    "secure": true,
    "changeOrigin":true,
    "logLevel": "debug"
  },
  {
    "context": ["/products"],
    "target": "https://fakestoreapi.com",
    "secure": true,
    "changeOrigin":true,
    "logLevel": "debug"
  }
]
```

Chạy ứng dụng ta có kết quả như mong muốn.

Bài 102: Tương tác External Restful API- Product Service 2**Yêu cầu:**

Tiếp tục cải tiến bài <https://fakestoreapi.com/products> ở trên. Cụ thể hiệu chỉnh giao diện kết quả như hình dưới đây:

**Hướng dẫn chi tiết:**

Ta thực hiện mọi bước giống như bài trước đã hướng dẫn. Chỉ cần bổ sung 2 công việc chính:

- Đầu tiên là cài đặt BootStrap JQuery (xem lại bước 5 của bài trước)
- Sau đó hiệu chỉnh “fake-product.component.html”

```
<p>fake-product works!</p>
<p>{{errMessage}}</p>
<h3 class="text-center text-white bg-dark p-3">Product List</h3>
<div class="container">
```

```
<div class="row">
    <div class="col-md-4" *ngFor="let p of data">
        <div class="card">
            <div class="img-container">
                <div class="d-flex justify-content-between align-items-center p-2 first">
                    <span class="percent">-25%</span>
                    <span class="wishlist"><i class="bi bi-heart-fill"></i></span>
                </div>
                <div class="text-center">
                    
                </div>
            </div>
            <div class="product-detail-container">
                <div class="d-flex justify-content-between align-items-center">
                    <h6 class="mb-0 text-truncate">{{p.title}}</h6>
                </div>
                <div>
                    <span class="text-danger fw-bold">${{p.price}}</span>
                </div>
                <div class="d-flex justify-content-between align-items-center mt-2">
                    <div class="ratings"><i class="bi bi-star-fill me-1"></i>
<span>{{p.rating.rate}}</span> </div>
                    <div class="count">Count:
{{p.rating.count}}</div>
                </div>
                <div class="mt-3"> <button class="btn btn-warning d-block w-100 text-white fw-bold">Buy Now</button> </div>
                </div>
            </div>
        </div>
    </div>
</div>
```

File CSS “fake-product.component.css”:

```
@import
url("https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;800&display=swap");
body {
    background-color: #d6d2d2;
    font-family: "Poppins", sans-serif;
    font-weight: 300
}

.card {
    border: none;
    border-radius: 10px
}

.card:hover {
    box-shadow: 5px 6px 6px 2px #cccecf;
    /* transform: scale(1.1) */
}
```

```
.percent {
    padding: 5px;
    background-color: #ffc107;
    border-radius: 5px;
    color: #fff;
    font-size: 14px;
    height: 25px;
    width: 60px;
    display: flex;
    justify-content: center;
    align-items: center;
    cursor: pointer
}

.wishlist {
    height: 30px;
    width: 30px;
    display: flex;
    justify-content: center;
    align-items: center;
    border-radius: 50%;
    background-color: #eee;
    padding: 10px;
    cursor: pointer
}

.img-container {
    position: relative
}

.img-container .first {
    position: absolute;
    width: 100%
}

.img-container img {
    border-top-left-radius: 5px;
    border-top-right-radius: 5px
}

.img-scale {
    width: 200px;
    height: 250px;
    object-fit: cover
}

.product-detail-container {
    padding: 10px
}

.ratings i {
    color: #a9a6a6
}

.ratings span {
    color: #a9a6a6
}

.count {
    color: #a9a6a6;
}

label.radio {
    cursor: pointer
}

label.radio input {
    position: absolute;
    top: 0;
```

```
left: 0;
visibility: hidden;
pointer-events: none
}
label.radio span {
height: 25px;
width: 25px;
display: flex;
justify-content: center;
align-items: center;
border: 2px solid #dc3545;
color: #dc3545;
font-size: 10px;
border-radius: 50%;
text-transform: uppercase
}
label.radio input:checked+span {
border-color: #dc3545;
background-color: #dc3545;
color: #fff
}
```

Chạy ứng dụng ta có kết quả như mong muốn.

Bài 103 – Coindesk API- Bitcoin Price Index in Real-time

Yêu cầu:

Sử dụng AngularJS để hiển thị Bitcoin Price Index (BPI) thời gian thực theo public API sau:

<https://api.coindesk.com/v1/bpi/currentprice.json>

```
{"time":{"updated":"Mar 12, 2023 10:43:00 UTC","updatedISO":"2023-03-12T10:43:00+00:00","updateduk":"Mar 12, 2023 at 10:43 GMT"},"disclaimer":"This data was produced from the CoinDesk Bitcoin Price Index (USD). Non-USD currency data converted using hourly conversion rate from openexchangerates.org","chartName":"Bitcoin","bpi":{"USD":{"code":"USD","symbol": "$","rate":20,520.4346,"description":"United States Dollar","rate_float":20520.4346}, "GBP":{"code":"GBP","symbol": "\u00a3","rate":17,146.711\u00a0,"description":"British Pound Sterling","rate_float":17146.711}, "EUR":{"code":"EUR","symbol": "\u20ac","rate":19,989.8993,"description":"Euro","rate_float":19989.8993}}}}
```

Hướng dẫn:

Bài 104 – Get List of Public APIs

Yêu cầu:

Viết chương trình hiển thị danh sách các public API ở link sau <https://api.publicapis.org/entries>, bố trí giao diện phù hợp.

Hướng dẫn:

Bài 105 - Predict the gender of a person based on their name - API.

Yêu cầu:

Thiết kế giao diện để sử dụng public API:

<https://api.genderize.io/?name=hoa>

API này nhận vào tên và dự đoán giới tính. Giao diện cần có ô nhập tên và nút kiểm tra giới tính.

Hướng dẫn:

Bài 106- Get US public data API

Yêu cầu:

Cho public US API:

<https://datausa.io/api/data?drilldowns=Nation&measures=Population>

Viết chương trình hiển thị dữ liệu lên giao diện, yêu cầu tạo cấu trúc model và bố trí giao diện phù hợp.

Hướng dẫn:

Bài 107- Random dog images API

Yêu cầu:

Cho public API: <https://dog.ceo/api/breeds/image/random>

Nó sẽ hiển thị ngẫu nhiên dữ liệu là link hình của một chú Dog, hãy viết chương trình hiển thị hình ảnh ngẫu nhiên này lên giao diện (mỗi lần nhấn vào nút hiển thị thì gọi hình ảnh trong API ngẫu nhiên này).

```
{"message": "https://images.dog.ceo/breeds/poodle-miniature/n02113712_9682.jpg", "status": "success"}
```



Hướng dẫn:

Module 11 –Restful API với NodeJS và ExpressJS

Nội dung kiến thức thực hành:

- + Thực hành mô hình Model-View-Controller (MVC)
- + Cài đặt và sử dụng ExpressJS
- + Cấu hình Webserver
- + Tạo các Restful API: GET, POST, PUT, DELETE
- + Sử dụng Postman để kiểm thử Restful API

Bài 108– Giải thích chi tiết mô hình hoạt động của MVC

Yêu cầu:

Giải thích chi tiết mô hình hoạt động của MVC, ý nghĩa của từng thành phần.

Bài 109 – API và Restful API

Yêu cầu:

API là gì? Hãy giải thích ý nghĩa của Restful API

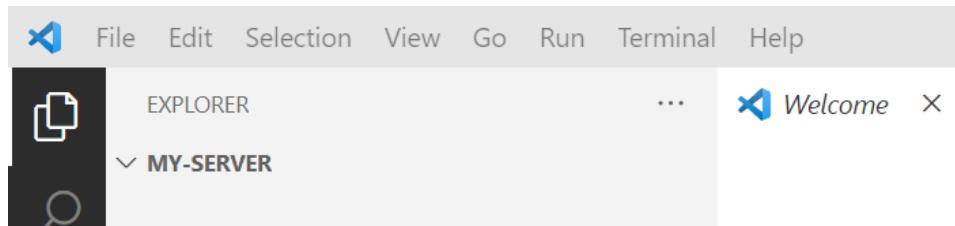
Bài 110 – Tao và thực thi Project Restful API

Yêu cầu:

Tạo một Project tên “my-server” để chạy trên nền tảng Web Server, sử dụng NodeJS và ExpressJS. Minh họa default API xuất thông báo “Hello Restful API”

Hướng dẫn:

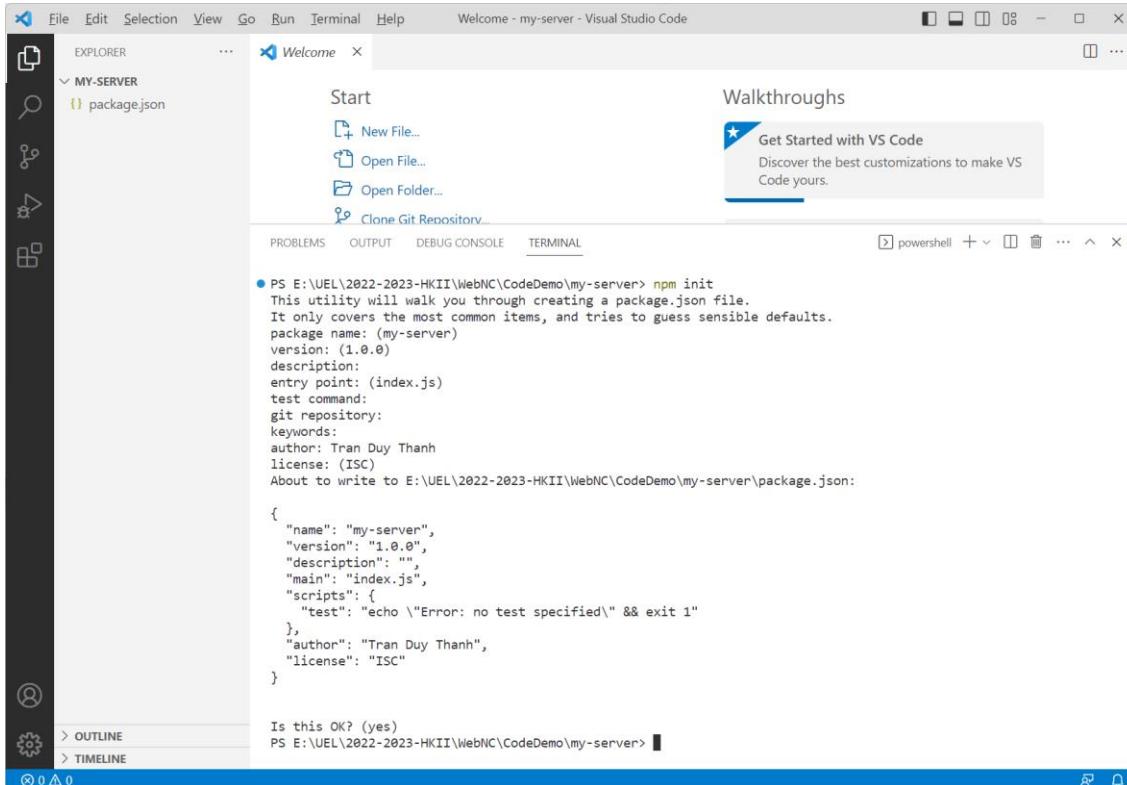
Bước 1: Tạo thư mục “my-server” và dùng Visual Studio Code để mở thư mục này:



Bước 2: Sau đó dùng terminal trong Visual Studio Code hoặc command line prompt để viết lệnh, hướng dẫn này sử dụng Terminal:

npm init

Minh họa quá trình chạy mã lệnh này:



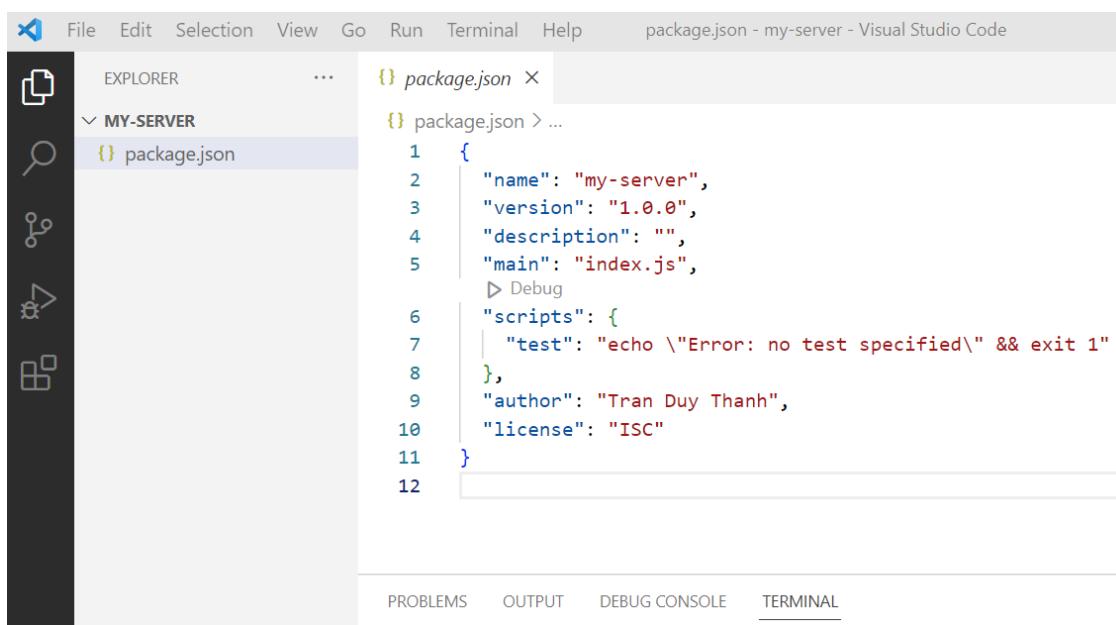
The screenshot shows the Visual Studio Code interface with the terminal tab selected. The terminal window displays the command `npm init` being run in a PowerShell-like environment. The output shows the creation of a package.json file with default settings. The file content is as follows:

```
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
package name: (my-server)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author: Tran Duy Thanh
license: (ISC)
About to write to E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server\package.json:

{
  "name": "my-server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "Tran Duy Thanh",
  "license": "ISC"
}

Is this OK? (yes)
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server>
```

Sau khi gõ `npm init` xong, thì cứ nhấn Enter liên tục (có thể cung cấp các thông tin ở từng dấu nhắc lệnh). Sau khi hoàn tất nó sẽ tạo ra một file **package.json**.



The screenshot shows the Visual Studio Code interface with the code editor tab selected. The editor window displays the contents of a package.json file. The file contains the following JSON object:

```
{
  "name": "my-server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "Tran Duy Thanh",
  "license": "ISC"
}
```

Bước 3: thực hiện lệnh: **npm install express** để cài đặt các thư viện ExpressJS

```
"main": "index.js",
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1"
},
"author": "Tran Duy Thanh",
"license": "ISC"
}

Is this OK? (yes)
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm install express
added 57 packages, and audited 58 packages in 3s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
○ PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server>
```

Sau khi thực hiện xong, thư viện express được cài đặt vào Project:

```
File Edit Selection View Go Run Terminal Help package.json - my-server - Visual Studio Code

EXPLORER ... {} package.json X
    MY-SERVER ...
        node_modules
        package-lock.json
        package.json
```

```
{} package.json > ...
1  {
2   "name": "my-server",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   ▷ Debug
7   "scripts": {
8     "test": "echo \"Error: no test specified\" && exit 1"
9   },
10  "author": "Tran Duy Thanh",
11  "license": "ISC",
12  "dependencies": {
13    "express": "^4.18.2"
14  }
15 }
```

Bước 4: Tạo một file “index.js” để tạo API mặc định như yêu cầu, coding như dưới đây:

```
File Edit Selection View Go Run Terminal Help
EXPLORER ... {} package.json JS index.js X
    MY-SERVER ...
        node_modules
        index.js
        package-lock.json
        package.json
```

```
{} package.json > ...
JS index.js > ...
1  const express=require("express")
2  const app=express()
3  const port=3000
4  //create default API
5  app.get("/",(req,res)=>{
6    res.send("Hello Restful API")
7  })
8  app.listen(port,()=>{
9    console.log(`My Server listening on port ${port}`)
10 })
```

Mã lệnh ở trên chạy dịch vụ Web (Restful API) ở port 3000, với API mặc định là xuất dữ liệu “Hello Restful API” khi nó được triệu gọi.

req(tên biến đại diện cho Request), là đối tượng để gửi các dữ liệu từ Client lên Server.

res(tên biến đại diện cho Response), là đối tượng để gửi các dữ liệu từ Server xuống Client.

Mã lệnh trên viết: **res.send**(“Hello Restful API”) tức là khi gọi mặc định dịch vụ web này thì nó trả về chuỗi “Hello Restful API”.

Bước 5: Khởi chạy Web Server:

Gọi lệnh **node index.js**

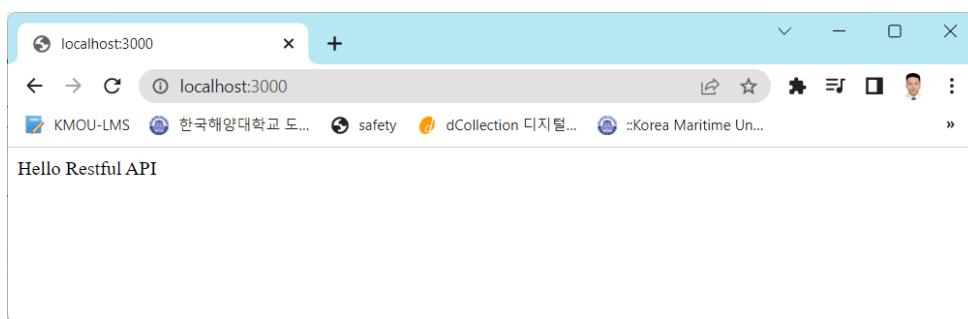
The screenshot shows the Visual Studio Code interface. The left sidebar has a tree view with 'MY-SERVER' expanded, showing 'node_modules', 'index.js', 'package-lock.json', and 'package.json'. The main area shows the 'index.js - my-server - Visual Studio Code' tab with the following code:

```
1 const express=require("express")
2 const app=express()
3 const port=3000
4 //create default API
5 app.get("/",(req,res)=>{
6   res.send("Hello Restful API")
7 })
8 app.listen(port,()=>{
9   console.log(`My Server listening on port ${port}`)
10 })
```

The terminal at the bottom shows the command being run: PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> node index.js and the output: My Server listening on port 3000.

Lúc này hàm listen sẽ được thực hiện, nếu Web Server được khởi tạo thành công thì lệnh ở dòng số 9 sẽ thực hiện, cụ thể ở đây là ta thấy thông báo “My Server listening on port 3000”.

Mở trình duyệt để xem kết quả:



Bài 111 – Cài đặt nodemon cho Web Server Project**Yêu cầu:**

Thông thường chúng ta phải thay đổi mã lệnh liên tục trong dự án, mỗi lần thay đổi thì phải chạy lại Web server cho dự án này, dẫn tới mất thời gian và không cần thiết. Kỹ thuật nodemon giúp restart lại server khi thay đổi mã lệnh trong dự án (phát âm **node-mon /nəʊd mən/**).

Hướng dẫn:**Bước 1: Thực hiện lệnh****npm i --save-dev nodemon**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm i --save-dev nodemon
added 32 packages, and audited 92 packages in 3s
10 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server>
```

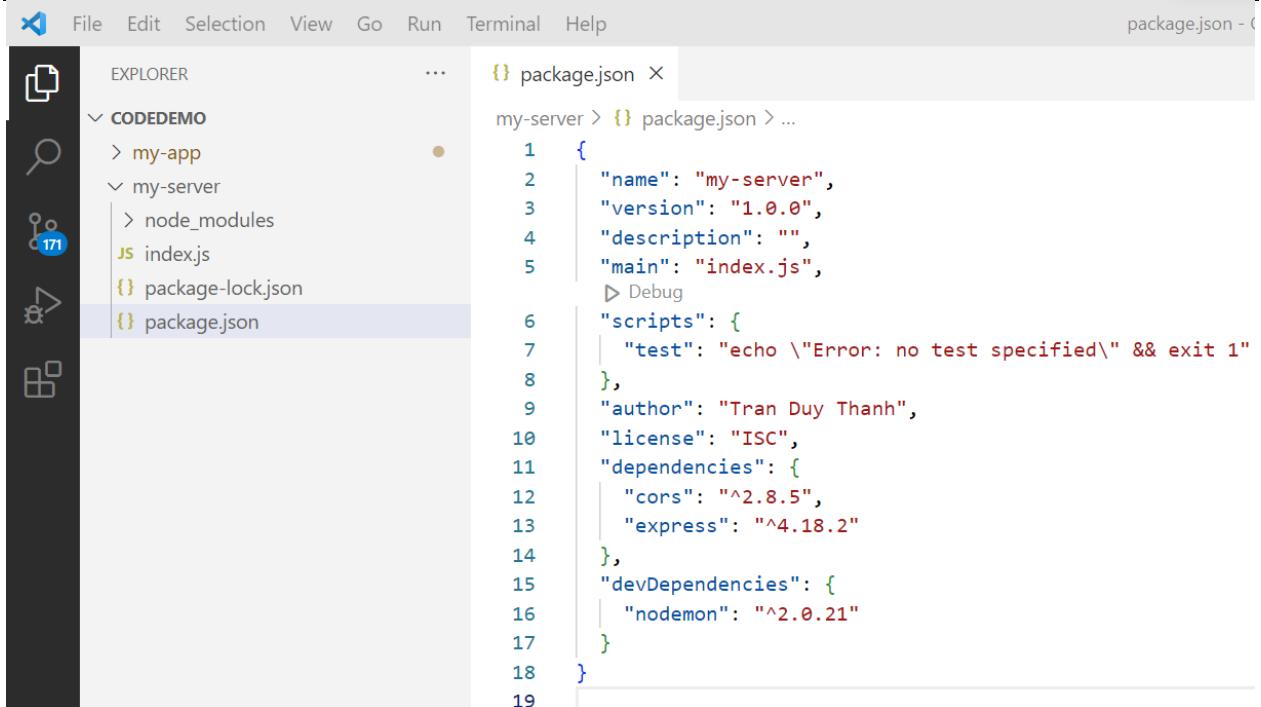
Bước 2: Thực hiện lệnh**npm i -g nodemon**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm i -g nodemon
changed 32 packages in 284ms
3 packages are looking for funding
  run `npm fund` for details
○ PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server>
```

Bước 3: Chính sửa “package.json”

Sau khi cài đặt xong các lệnh nodemon thì file “package.json” sẽ như hình chụp dưới đây:

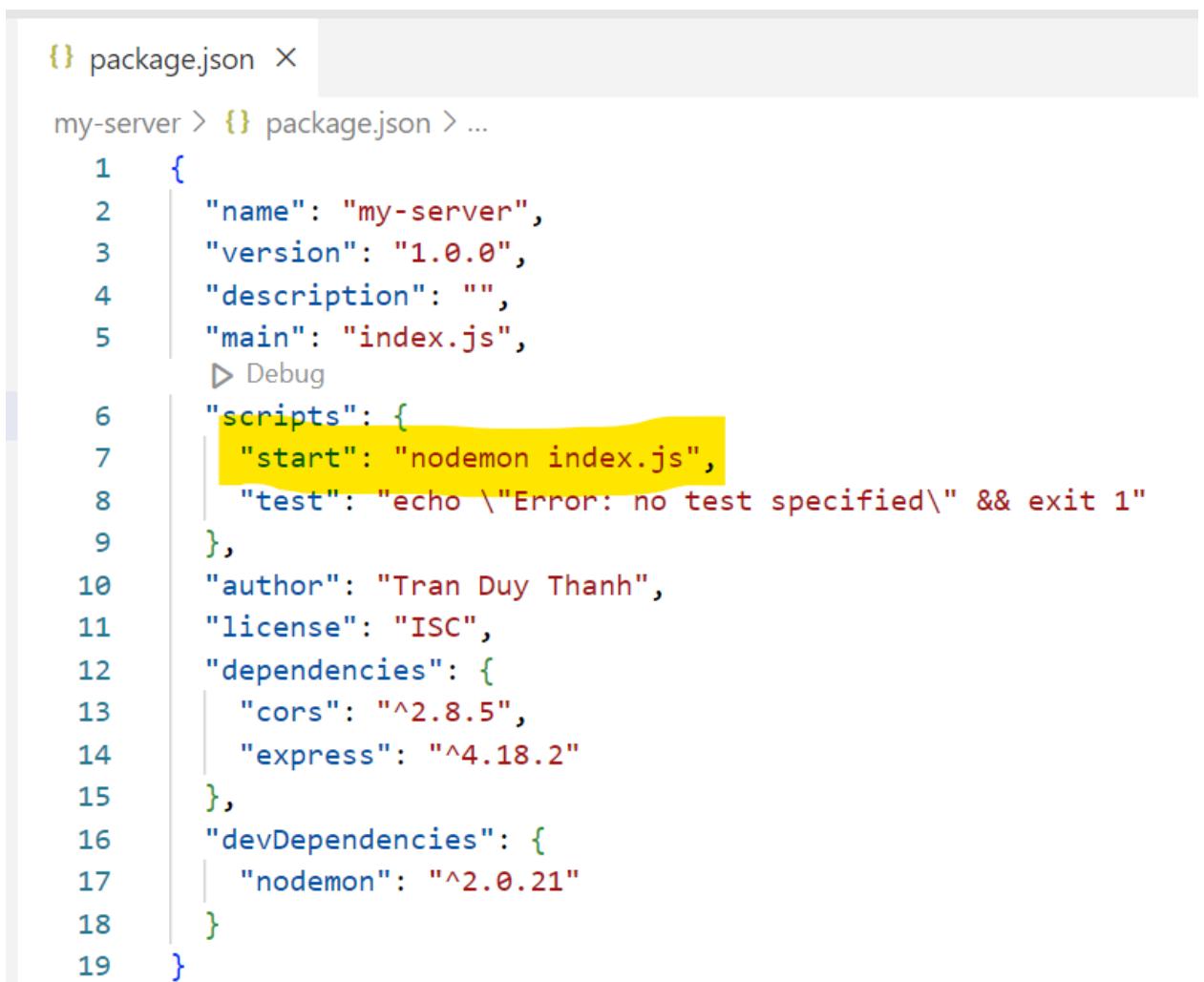


```
File Edit Selection View Go Run Terminal Help package.json - 0

EXPLORER ... package.json X
CODEDEMO my-server > package.json ...
my-app my-server > node_modules
index.js package-lock.json package.json

1 {
2   "name": "my-server",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   ▶ Debug
7   "scripts": {
8     "test": "echo \\\"Error: no test specified\\\" && exit 1"
9   },
10  "author": "Tran Duy Thanh",
11  "license": "ISC",
12  "dependencies": {
13    "cors": "^2.8.5",
14    "express": "^4.18.2"
15  },
16  "devDependencies": {
17    "nodemon": "^2.0.21"
18  }
19 }
```

Bổ sung lệnh “start”: “nodemon index.js” vào thẻ “scripts”



```
package.json X
my-server > package.json ...
1 {
2   "name": "my-server",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   ▶ Debug
7   "scripts": {
8     "start": "nodemon index.js",
9     "test": "echo \\\"Error: no test specified\\\" && exit 1"
10    },
11    "author": "Tran Duy Thanh",
12    "license": "ISC",
13    "dependencies": {
14      "cors": "^2.8.5",
15      "express": "^4.18.2"
16    },
17    "devDependencies": {
18      "nodemon": "^2.0.21"
19    }
}
```

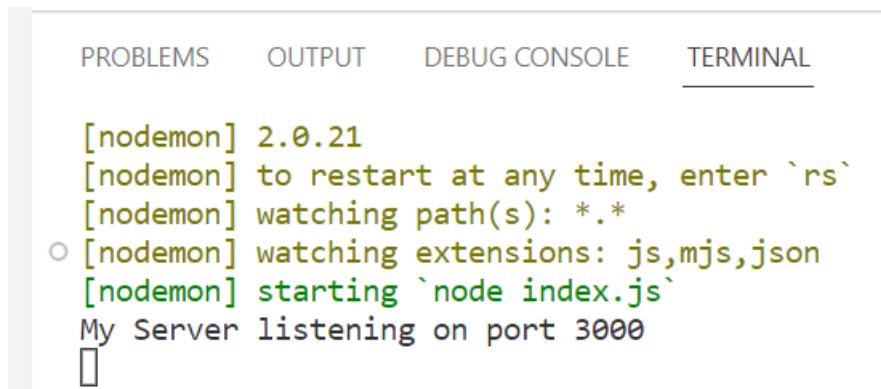
Bước 4: Thực hiện lại lệnh chạy Web Server cho API này.

Thay vì viết

node index.js

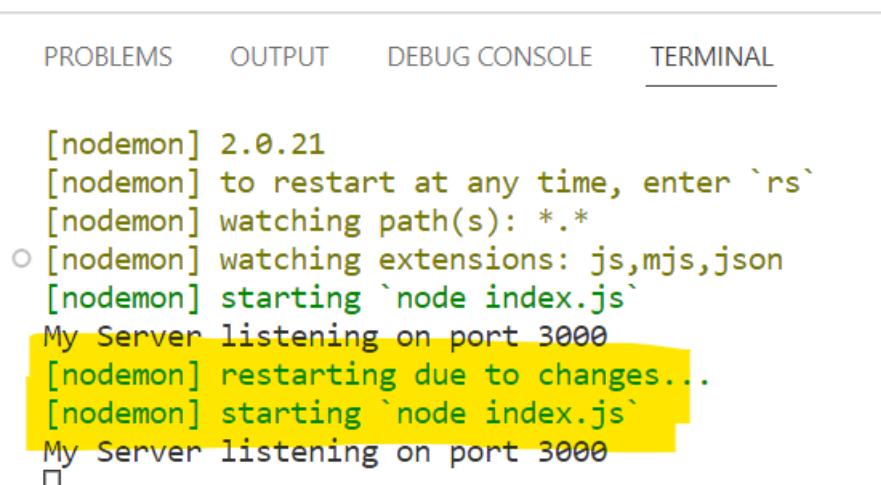
thì ta viết:

npm start



```
[nodemon] 2.0.21
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
My Server listening on port 3000
□
```

Như vậy Server đã được start ở port 3000 với cơ chế nodemon để tự động restart khi mã nguồn thay đổi, ví dụ nếu ta thay đổi bất kỳ mã nguồn:



```
[nodemon] 2.0.21
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
My Server listening on port 3000
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
My Server listening on port 3000
□
```

Bài 112 – Cài đặt morgan cho Web Server Project

Yêu cầu:

Ngoài ra ta có thể dùng lệnh morgan để xem http request logger

Hướng dẫn:

Bước 1: Thực hiện lệnh

npm i --save-dev morgan

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
● PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm i --save-dev morgan
added 5 packages, and audited 97 packages in 4s
10 packages are looking for funding
  run `npm fund` for details
○ found 0 vulnerabilities
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> 
```

File “package.json” sẽ được cập nhật:

```
{} package.json X JS index.js
my-server > {} package.json > ...
1  {
2    "name": "my-server",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    ▶ Debug
7    "scripts": {
8      "start": "nodemon index.js",
9      "test": "echo \"Error: no test specified\" && exit 1"
10     },
11    "author": "Tran Duy Thanh",
12    "license": "ISC",
13    "dependencies": {
14      "cors": "^2.8.5",
15      "express": "^4.18.2"
16    },
17    "devDependencies": {
18      "morgan": "^1.10.0",
19      "nodemon": "^2.0.21"
20    }
}
```

Bước 2: Chỉnh sửa index.js để sử dụng morgan

```
const express=require("express")
const app=express()
const port=3000
const morgan=require("morgan")
app.use(morgan("combined"))
//create default API
```

```
app.get("/",(req,res)=>{
    res.send("Hello Restful API")
})
app.listen(port,()=>{
    console.log(`My Server listening on port ${port}`)
})
```

Bước 3: Khởi chạy Server và ta có kết quả http request logger như hình bên dưới:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
0 found vulnerabilities
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm start
> my-server@1.0.0 start
> nodemon index.js

[nodemon] 2.0.21
[nodemon] to restart at any time, enter `rs`
[nodemon] starting `node index.js`
My Server listening on port 3000
::1 - - [08/Mar/2023:16:32:12 +0000] "GET /books/ HTTP/1.1" 304 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/111.0.0.0 Safari/537.36"
::1 - - [08/Mar/2023:16:32:26 +0000] "GET /books/ HTTP/1.1" 200 433 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.63"
::1 - - [08/Mar/2023:16:32:26 +0000] "GET /favicon.ico HTTP/1.1" 404 150 "http://localhost:3000/books/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.63"
```

Bài 113 – Tao HTTP GET – List of Book

Yêu cầu:

Trong file index.js, tiếp tục tạo một danh sách Sách, viết API để lấy danh sách này.

Hướng dẫn:

Giả sử trong file index.js ta bổ sung khai báo danh sách Sách như dưới đây:

```
let database=[
    {"BookId":"b1","BookName":"Kỹ thuật lập trình cơ bản",
    "Price":70,"Image":"b1.png"},
    {"BookId":"b2","BookName":"Kỹ thuật lập trình nâng cao",
    "Price":100,"Image":"b2.png"},
    {"BookId":"b3","BookName":"Máy học cơ bản","Price":200,"Image":"b3.png"},
    {"BookId":"b4","BookName":"Máy học nâng cao","Price":300,"Image":"b4.png"},
    {"BookId":"b5","BookName":"Lập trình Robot cơ bản","Price":250,"Image":"b5.png"},]
```

Để tạo API lấy danh sách Sách thì thường ta khai báo pattern “/books” như sau:

```
app.get("/books",(req,res)=>{
    res.send(database)
})
```

Mã lệnh đầy đủ của chương trình (index.js) sau khi bổ sung API này là:

```
const express=require("express")
const app=express()
const port=3000
const morgan=require("morgan")
app.use(morgan("combined"))

app.get("/",(req,res)=>{
    res.send("Hello Restful API")
})
app.listen(port,()=>{
    console.log(`My Server listening on port ${port}`)
})
let database=[
    {"BookId":"b1","BookName":"Kỹ thuật lập trình cơ bản","Price":70,"Image":"b1.png"},  

    {"BookId":"b2","BookName":"Kỹ thuật lập trình nâng cao","Price":100,"Image":"b2.png"},  

    {"BookId":"b3","BookName":"Máy học cơ bản","Price":200,"Image":"b3.png"},  

    {"BookId":"b4","BookName":"Máy học nâng cao","Price":300,"Image":"b4.png"},  

    {"BookId":"b5","BookName":"Lập trình Robot cơ bản","Price":250,"Image":"b5.png"},  

]
app.get("/books",(req,res)=>{
    res.send(database)
})
```

Tiến hành chạy lại lệnh: **node index.js** (hoặc nếu trước đó đã chạy **npm start** rồi thì hệ thống tự động restart, ta không phải chạy lại) và triệu gọi api books ta sẽ có kết quả:



Bài 114 - Triệu gọi HTTP GET – List of Book**Yêu cầu:**

Từ API **books** được tạo ở trên. Hãy viết chương trình triệu gọi API này trong AngularJS.

Coi như trong assets/book_images có 5 hình có tên như trong Server trả về. giao diện sau khi triệu gọi xong API books:

BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	

Hướng dẫn:

*Trong project “**my-server**”. Thông thường ta sẽ gặp báo lỗi CORS Policy, nên trên Server ta bổ sung các lệnh sau:

Bước 1: cài đặt lệnh cors(Cross-Origin Resource Sharing) cho Web server có chưa api books.

npm i cors

Bước 2: Bổ sung thêm lệnh liên quan tới cors, mã lệnh cuối cùng của index.js:

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... JS index.js ...
my-server > JS index.js > ...
1 const express=require("express")
2 const app=express()
3 const port=3000
4 const morgan=require("morgan")
5 app.use(morgan("combined"))
6 //create default API
7 app.get("/",(req,res)=>{
8   res.send("Hello Restful API")
9 })
10 app.listen(port,()=>{
11   console.log(`My Server listening on port ${port}`)
12 })
13
14 const cors=require("cors")
15 app.use(cors())
16
17 let database=[
18   {"BookId":"b1","BookName":"Kỹ thuật lập trình cơ bản","Price":70,"Image":"b1.png"}, 
19   {"BookId":"b2","BookName":"Kỹ thuật lập trình nâng cao","Price":100,"Image":"b2.png"}, 
20   {"BookId":"b3","BookName":"Máy học cơ bản","Price":200,"Image":"b3.png"}, 
21   {"BookId":"b4","BookName":"Máy học nâng cao","Price":300,"Image":"b4.png"}, 
22   {"BookId":"b5","BookName":"Lập trình Robot cơ bản","Price":250,"Image":"b5.png"}, 
23 ]
24
25 app.get("/books",cors(),(req,res)=>{
26   res.send(database)
27 })

```

Dòng lệnh 14, 15 khai báo thư viện cors và enable cors

Dòng lệnh 25 bổ sung thêm cors() cho api books

*Trong project “**my-app**”:

Bước 1: Tạo interface Ibook trong file Book.ts

```

export interface IBook{
  BookId:string,
  BookName:string,
  Price:number,
  Image:string
}

```

Bước 2: Tạo một service mang tên “BookAPI”, trong file “book-api.service.ts”
viết các lệnh sau:

```

import { HttpClient, HttpErrorResponse, HttpHeaders } from
'@angular/common/http';
import { Injectable } from '@angular/core';
import { catchError, map, Observable, retry, throwError } from 'rxjs';
import { IBook } from './interfaces/Book';

@Injectable({
  providedIn: 'root'
})
export class BookAPIService {

  constructor(private _http: HttpClient) { }

  getBooks():Observable<any>
}

```

```
{  
    const headers=new HttpHeaders().set("Content-Type","text/plain;charset=utf-  
8")  
    const requestOptions:Object={  
        headers:headers,  
        responseType:"text"  
    }  
    return this._http.get<any>("/books",requestOptions).pipe(  
        map(res=>JSON.parse(res) as Array<IBook>),  
        retry(3),  
        catchError(this.handleError)  
    )  
  
    handleError(error:HttpErrorResponse){  
        return throwError(()=>new Error(error.message))  
    }  
}
```

Bước 3: Cấu hình “proxy.conf.json”, tiếp tục bổ sung vào cuối file:

```
[  
{  
    "context": ["/exchange"],  
    "target": "https://www.dongabank.com.vn",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
},  
{  
    "context": ["/products"],  
    "target": "https://fakestoreapi.com",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
},  
{  
    "context": ["/books"],  
    "target": "http://localhost:3000",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
}  
]
```

Bước 4: Tạo một component “Books”.

Trong file “books.component.ts” bổ sung lệnh:

```
import { Component } from '@angular/core';
import { BookAPIService } from '../book-api.service';

@Component({
  selector: 'app-books',
  templateUrl: './books.component.html',
  styleUrls: ['./books.component.css']
})
export class BooksComponent {
  books: any;
  errorMessage: string = '';
  constructor(private _service: BookAPIService) {
    this._service.getBooks().subscribe({
      next: (data) => {this.books = data},
      error: (err) => {this.errorMessage = err}
    })
  }
}
```

Trong file “books.component.html” bổ sung lệnh:

```
<p>books works!</p>
{{errorMessage}}
<table border="1">
  <tr>
    <td>BookId</td>
    <td>BookName</td>
    <td>Price</td>
    <td>Image</td>
  </tr>
  <tbody>
    <tr *ngFor="let book of books">
      <td>{{book.BookId}}</td>
      <td>{{book.BookName}}</td>
      <td>{{book.Price}}</td>
      <td>
        
      </td>
    </tr>
  </tbody>
</table>
```

Bước 5: Khai báo Routing hoặc tham chiếu trực tiếp trong “app.component.html”, chạy ứng dụng “my-app” ta có kết quả như đề bài:

The screenshot shows a web browser window titled "MyApp" at "localhost:4200/books". The page displays a table of books with columns: BookId, BookName, Price, and Image. The data is as follows:

BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	

Như vậy dữ liệu trong books API được xây dựng trong “my-server” đã được nạp lên front end Angularjs “my-app” thành công.

Bài 115 – Tao HTTP GET – a Book

Yêu cầu:

Xây dựng API lấy chi tiết thông tin một Book khi biết BookId. Ví dụ

<http://localhost:3000/books/b1>

sẽ trả về thông tin chi tiết của Book có BookId=b1

<http://localhost:3000/books/b2>

sẽ trả về thông tin chi tiết của Book có BookId=b2

ví dụ khi nhập b1, thì JsonObject của Book sẽ cho ra kết quả:

The screenshot shows a web browser window at "localhost:3000/books/b1". The page displays the following JSON object:

```
{"BookId": "b1", "BookName": "Kỹ thuật lập trình cơ bản", "Price": 70, "Image": "b1.png"}
```

Hướng dẫn:

Trong “index.js” bổ sung lệnh sau vào cuối tập tin:

```
app.get("/books/:id", cors(), (req, res)=>{
    id=req.params["id"]
    let p=database.find(x=>x.BookId==id)
    res.send(p)
})
```

Mã lệnh đầy đủ:

```
const express=require("express")
const app=express()
const port=3000
const morgan=require("morgan")
app.use(morgan("combined"))
//create default API
app.get("/",(req,res)=>{
    res.send("Hello Restful API")
})
app.listen(port,()=>{
    console.log(`My Server listening on port ${port}`)
})

const cors=require("cors")
app.use(cors())

let database=[
    {"BookId":"b1","BookName":"Kỹ thuật lập trình cơ bản","Price":70,"Image":"b1.png"}, 
    {"BookId":"b2","BookName":"Kỹ thuật lập trình nâng cao","Price":100,"Image":"b2.png"}, 
    {"BookId":"b3","BookName":"Máy học cơ bản","Price":200,"Image":"b3.png"}, 
    {"BookId":"b4","BookName":"Máy học nâng cao","Price":300,"Image":"b4.png"}, 
    {"BookId":"b5","BookName":"Lập trình Robot cơ bản","Price":250,"Image":"b5.png"}, 
]

app.get("/books",cors(),(req,res)=>{
    res.send(database)
})
app.get("/books/:id",cors(),(req,res)=>{
    id=req.params["id"]
    let p=database.find(x=>x.BookId==id)
    res.send(p)
})
```

Lưu ý cú pháp:

"/books/:id"

Bài 116– Triệu gọi HTTP GET – a Book**Yêu cầu:**

Xây dựng giao diện bên AngularJS, cho phép nhập vào BookId để tìm kiếm thông tin chi tiết của Book này bằng cách triệu gọi API Get a Book ở bài trước.

**Hướng dẫn:**

Bước 1: Tiếp tục mở lại file “book-api.service.ts” để hiệu chỉnh mã nguồn cho lớp “BookAPIService” service.

```
getBook(bookId:string):Observable<any>
{
  const headers=new HttpHeaders().set("Content-Type","text/plain;charset=utf-8")
  const requestOptions:Object={
    headers:headers,
    responseType:"text"
  }
  return this._http.get<any>("/books/"+bookId,requestOptions).pipe(
    map(res=>JSON.parse(res) as IBook),
    retry(3),
    catchError(this.handleError))
}
```

Mã lệnh đầy đủ của Service này sau khi cập nhật hàm getBook(bookId:string)

```
import { HttpClient, HttpErrorResponse, HttpHeaders } from
'@angular/common/http';
import { Injectable } from '@angular/core';
import { catchError, map, Observable, retry, throwError } from 'rxjs';
```

```
import { IBook } from './interfaces/Book';

@Injectable({
  providedIn: 'root'
})
export class BookAPIService {

  constructor(private _http: HttpClient) { }

  getBooks():Observable<any>
  {
    const headers=new HttpHeaders().set("Content-Type", "text/plain;charset=utf-8")
    const requestOptions:Object={
      headers:headers,
      responseType:"text"
    }
    return this._http.get<any>("/books",requestOptions).pipe(
      map(res=>JSON.parse(res) as Array<IBook>),
      retry(3),
      catchError(this.handleError)
    )
  }

  handleError(error:HttpErrorResponse){
    return throwError(()=>new Error(error.message))
  }

  getBook(bookId:string):Observable<any>
  {
    const headers=new HttpHeaders().set("Content-Type", "text/plain;charset=utf-8")
    const requestOptions:Object={
      headers:headers,
      responseType:"text"
    }
    return this._http.get<any>("/books/"+bookId,requestOptions).pipe(
      map(res=>JSON.parse(res) as IBook),
      retry(3),
      catchError(this.handleError)
    )
  }
}
```

Bước 2: Tạo component “BookDetail” để hiển thị thông tin chi tiết của Book

File “book-detail.component.ts” bổ sung mã lệnh:

```
import { Component } from '@angular/core';
import { BookAPIService } from '../book-api.service';

@Component({
  selector: 'app-book-detail',
  templateUrl: './book-detail.component.html',
  styleUrls: ['./book-detail.component.css']
})
export class BookDetailComponent {
  book: any;
  errorMessage: string = '';
  constructor(private _service: BookAPIService) {}

  searchBook(bookId: string) {
    this._service.getBook(bookId).subscribe({
      next: (data) => {this.book = data},
      error: (err) => {this.errorMessage = err}
    })
  }
}
```

File “book-detail.component.html” bổ sung mã lệnh:

```
<p>book-detail works!</p>
<p>{{errorMessage}}</p>
Book Id: <input type="text" #bookId> <button
(click)="searchBook(bookId.value)">Search</button>
<table>
  <tr>
    <td>Book Id:</td>
    <td>{{book.BookId}}</td>
  </tr>
  <tr>
    <td>Book Name:</td>
    <td>{{book.BookName}}</td>
  </tr>
  <tr>
    <td>Price:</td>
    <td>{{book.Price}}</td>
  </tr>
  <tr>
    <td colspan="2">
      
    </td>
  </tr>
</table>
```

Khai báo routing hoặc tham chiếu trực tiếp component này trong “app.component.html” để test. Chạy ứng dụng ta có kết quả như yêu cầu.

Bài 117 – Tao HTTP POST – Create a Book**Yêu cầu:**

Bổ sung thêm API cho phép thêm mới một Book.

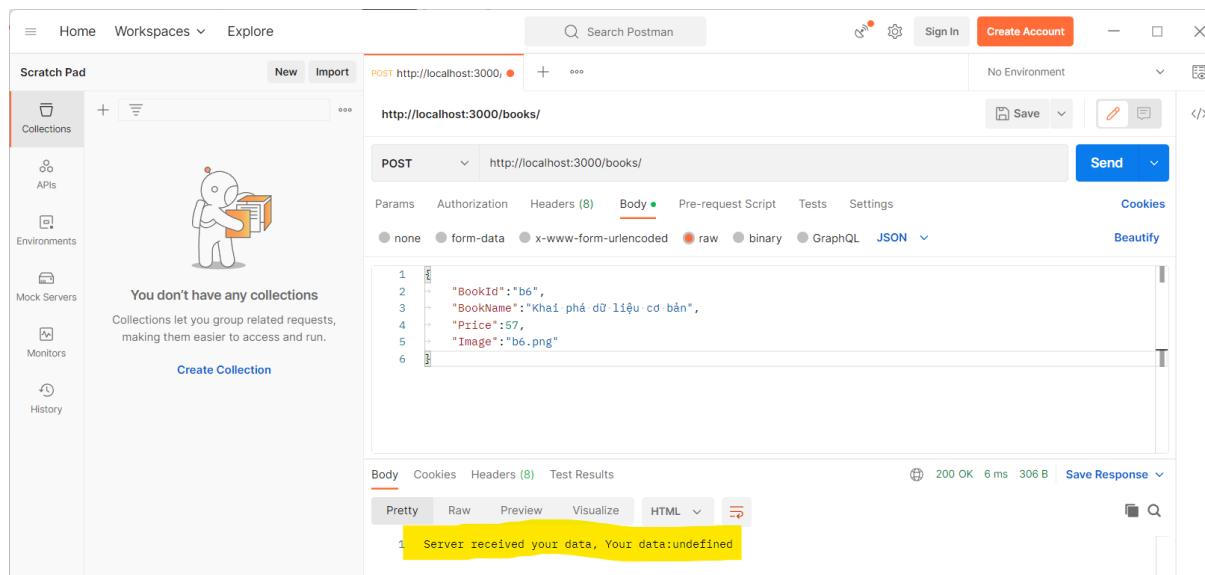
Hướng dẫn:

Bước 1: trong dự án “my-server”, tiếp tục bổ sung mã lệnh sau vào cuối tập tin “index.js”

```
app.post("/books", cors(), (req, res) => {
    console.log(req.body)
    res.send("Server received your data, Your data:" + req.body)
})
```

Bước 2: Cài đặt Postman và sử dụng để test các method: POST, PUT, DELETE (vì chỉ có GET mới test được trên trình duyệt). Lưu ý rằng nếu Postman test thành công thì xem như các lệnh API ở “my-server” đã viết đúng. Công cụ này rất tiện lợi, nó giúp ta kiểm thử API trước khi sử dụng. Ngoài ra còn có công cụ Swagger(<https://swagger.io/>). Trong nội dung của môn học này chúng ta sẽ sử dụng Postman. Tải ở đây: <https://www.postman.com/downloads/>

Test thử nghiệm api books với method POST, cấu hình như bên dưới, nhấn send.



Rõ ràng thấy đã triệu gọi API books (method POST) thành công nhưng body chưa lấy được (undefined).

Bước 3: Cài đặt body-parser cho “my-server”

npm i body-parser

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
● PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> npm i body-parser
added 2 packages, changed 2 packages, and audited 99 packages in 1s
10 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
○ PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server> 
```

sau khi cài thư viện xong thì tiến hành chỉnh sửa **index.js**:

```
const bodyParser=require("body-parser")
app.use(bodyParser.json())
```

Bước 4: Chính sửa lại API post a book

```
app.post("/books",cors(),(req,res)=>{
  //put json book into database
  database.push(req.body);
  //send message to client(send all database to client)
  res.send(database)
})
```

Với lệnh ở trên, mỗi lần POST 1 Book lên server nó sẽ lưu vào database. Sau đó ta có thể gọi GET books để xem lại toàn bộ dữ liệu mới nhất. Đây là minh họa thao tác thêm mới 1 đối tượng lên Server, nó sẽ bị mất khi restart server. Do đó ta thường phải lưu vào Cơ sở dữ liệu là vậy, chương trình học sẽ dùng MongoDB để lưu trữ, sẽ được hướng dẫn ở các module sau.

Mã lệnh đầy đủ của index.js sau khi bổ sung API POST 1 Book:

```
const express=require("express")
const app=express()
const port=3000
const morgan=require("morgan")
app.use(morgan("combined"))
const bodyParser=require("body-parser")
app.use(bodyParser.json())
//create default API
app.get("/",(req,res)=>{
  res.send("Hello Restful API")
})
app.listen(port,()=>{
  console.log(`My Server listening on port ${port}`)
})

const cors=require("cors")
app.use(cors())
```

```

let database=[  
    {"BookId":"b1","BookName":"Kỹ thuật lập trình cơ  
bản","Price":70,"Image":"b1.png"},  
    {"BookId":"b2","BookName":"Kỹ thuật lập trình nâng  
cao","Price":100,"Image":"b2.png"},  
    {"BookId":"b3","BookName":"Máy học cơ bản","Price":200,"Image":"b3.png"},  
    {"BookId":"b4","BookName":"Máy học nâng cao","Price":300,"Image":"b4.png"},  
    {"BookId":"b5","BookName":"Lập trình Robot cơ bản","Price":250,"Image":"b5.png"},  
]  
app.get("/books",cors(),(req,res)=>{  
    res.send(database)  
})  
app.get("/books/:id",cors(),(req,res)=>{  
    id=req.params["id"]  
    let p=database.find(x=>x.BookId==id)  
    res.send(p)  
})  
app.post("/books",cors(),(req,res)=>{  
    //put json book into database  
    database.push(req.body);  
    //send message to client(send all database to client)  
    res.send(database)  
})

```

Bước 5: thử nghiệm lại bằng Postman

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Scratch Pad' and various sections like Collections, APIs, Environments, Mock Servers, Monitors, and History. The main area has a search bar at the top right. Below it, a POST request is being made to 'http://localhost:3000/books'. The 'Body' tab is selected, showing a JSON input field with the following data:

```

1 "BookId": "b6",
2 "BookName": "Khai phá dữ liệu cơ bản",
3 "Price": 55,
4 "Image": "b6.png"

```

Below the JSON input, the response is shown in 'Pretty' format:

```

30     "Image": "b6.png"
31   },
32   {
33     "BookId": "b6",
34     "BookName": "Khai phá dữ liệu cơ bản",
35     "Price": 55,
36     "Image": "b6.png"
37   }
38 ]

```

The response status is 200 OK with 10 ms and 791 B.

Chọn method là POST, thẻ body chọn raw/json ta nhập dữ liệu có cấu trúc:

```
{
    "BookId": "b6",
    "BookName": "Khai phá dữ liệu cơ bản",
    "Price": 55,
    "Image": "b6.png"
}
```

Sau đó bấm nút “Send”, nó sẽ triệu gọi API Post a Book, nếu thành công nó sẽ đồng thời trả về toàn bộ dữ liệu mới nhất (cho chúng ta test), dĩ nhiên ta chỉ cần trả về true/false là được rồi. Trong ví dụ này tài liệu trả về toàn bộ Book để sinh viên dễ quan sát. Sinh viên có thể test lại bằng API GET all Books đã trình bày ở bài trước. Ví dụ chỉ cần nhấn lại chức năng xem Books trên giao diện Web, nó sẽ tải lại dữ liệu mới nhất:

books works!

BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	
b6	Khai phá dữ liệu cơ bản	55	

Dữ liệu đã được hiển thị cho sách có mã {b6}, hình ảnh ta chưa xử lý, qua phần MongoDB ta sẽ xử lý hình ảnh.

Bài 118 – Triệu gọi HTTP POST – Create a Book

Yêu cầu:

Thiết kế giao diện Front End (my-app) để nhập dữ liệu Book rồi triệu gọi API ở bài trước để tạo mới 1 Book:

- Mặc định khởi động màn hình sẽ tải toàn bộ Books ở Server về (Sinh viên có thể tái sử dụng **BooksComponent** hoặc lặp lại thao tác triệu gọi API lấy toàn bộ Books)
- Mỗi lần submit thành công 1 Book thì màn hình sẽ cập nhật lại danh sách.

Book Id:	<input type="text" value="Nhập Book Id"/>		
Book Name:	<input type="text" value="Nhập Book Name"/>		
Price:	<input type="text" value="0"/>		
<input type="button" value="Submit"/>			
BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	

Hướng dẫn:

Bước 1: Trong tập tin “Book.ts”, bổ sung lớp Book:

```
export interface IBook{
    BookId:string,
    BookName:string,
    Price:number,
    Image:string
}
export class Book{
    constructor(
        public BookId:string="",
        public BookName:string="",
        public Price:number=0,
        public Image:string="")
    {}
}
```

Bước 2: Trong tập tin “book-api.service.ts”, Bổ sung mã lệnh cho BookAPIService:

```
postBook(aBook:any):Observable<any>
{
    const headers=new HttpHeaders().set("Content-Type","application/json;charset=utf-8")
```

```
const requestOptions: Object = {
  headers: headers,
  responseType: "text"
}
return
this._http.post<any>("/books", JSON.stringify(aBook), requestOptions).pipe(
  map(res => JSON.parse(res) as Array<IBook>),
  retry(3),
  catchError(this.handleError))
}
```

Vì ở my-server ta nhận json nên content-type chính là **application/json**.

aBook là 1 model (Book) nhận từ giao diện, nó cần được đưa về định dạng Json as String bằng lệnh **JSON.stringify(aBook)** để gửi lệnh Server (API HTTP POST)
mã lệnh đầy đủ của BookAPIService sau khi chỉnh sửa:

```
import { HttpClient, HttpHeaders, HttpHeaders } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { catchError, map, Observable, retry, throwError } from 'rxjs';
import { IBook } from './interfaces/Book';

@Injectable({
  providedIn: 'root'
})
export class BookAPIService {

  constructor(private _http: HttpClient) { }

  getBooks(): Observable<any>
  {
    const headers = new HttpHeaders().set("Content-Type", "text/plain; charset=utf-8")
    const requestOptions: Object = {
      headers: headers,
      responseType: "text"
    }
    return this._http.get<any>("/books", requestOptions).pipe(
      map(res => JSON.parse(res) as Array<IBook>),
      retry(3),
      catchError(this.handleError))
  }
  handleError(error: HttpErrorResponse){
    return throwError(() => new Error(error.message))
  }
  getBook(bookId: string): Observable<any>
  {
    const headers = new HttpHeaders().set("Content-Type", "text/plain; charset=utf-8")
    const requestOptions: Object = {
      headers: headers,
      responseType: "text"
    }
  }
}
```

```
        return this._http.get<any>("/books/" + bookId, requestOptions).pipe(
            map(res=>JSON.parse(res) as IBook),
            retry(3),
            catchError(this.handleError))
    }
    postBook(aBook:any):Observable<any>
    {
        const headers=new HttpHeaders().set("Content-Type","application/json;charset=utf-8")
        const requestOptions:object={
            headers:headers,
            responseType:"text"
        }
        return this._http.post<any>("/books",JSON.stringify(aBook),requestOptions).pipe(
            map(res=>JSON.parse(res) as Array<IBook>),
            retry(3),
            catchError(this.handleError))
    }
}
```

Bước 3: Tạo thêm BookNewComponent.

Chỉnh sửa mã lệnh trong tập tin “**book-new.component.ts**”:

```
import { Component } from '@angular/core';
import { FormControl, FormGroup } from '@angular/forms';
import { BookAPIService } from '../book-api.service';
import { Book } from '../interfaces/Book';

@Component({
    selector: 'app-book-new',
    templateUrl: './book-new.component.html',
    styleUrls: ['./book-new.component.css']
})
export class BookNewComponent {
    book=new Book();
    books:any
    errMessage:string=''
    constructor(private _service: BookAPIService){
        this._service.getBooks().subscribe({
            next:(data)=>{this.books=data},
            error:(err)=>{this.errMessage=err}
        })
    }
    postBook()
    {
        this._service.postBook(this.book).subscribe({
            next:(data)=>{this.books=data},
            error:(err)=>{this.errMessage=err}
        })
    }
}
```

-Biến **book** là biến dùng để mapping theo ngModel. Khi người dùng nhập dữ liệu trên giao diện nó sẽ tự động mapping các thuộc tính và ô nhập liệu.

-Biến **books** để tải toàn bộ book trên server nhằm phục vụ cho việc hiển thị lại danh sách.

-Để mặc định khi mở giao diện thì danh sách Book được hiển thị thì trong constructor ta triệu gọi lại API HTTP GET books (ở đây nếu ta tái sử dụng BookComponent thì không cần gọi lại)

-Hàm postBook() sẽ triệu gọi API POST Book để đẩy dữ liệu từ Giao diện lên Server để lưu trữ. Nếu thực hiện thành công nó sẽ trả về toàn bộ dữ liệu mới nhất, lúc này ta cập nhật lại danh sách.

Chỉnh sửa mã lệnh trong tập tin “**book-new.component.html**”:

```
<p>book-new works!</p>
<p>{{errMessage}}</p>
<table>
  <tr>
    <td>Book Id:</td>
    <td>
      <input type="text" class="form-control" #name="ngModel" name="BookId"
id="BookId" placeholder="Nhập Book Id"
[(ngModel)]="book.BookId"/>
    </td>
  </tr>
  <tr>
    <td>Book Name:</td>
    <td>
      <input type="text" class="form-control" #name="ngModel"
name="BookName" id="BookName" placeholder="Nhập Book Name"
[(ngModel)]="book.BookName"/>
    </td>
  </tr>
  <tr>
    <td>Price:</td>
    <td>
      <input type="text" class="form-control" #name="ngModel"
name="BookPrice" id="BookPrice" placeholder="Nhập Price"
[(ngModel)]="book.Price"/>
    </td>
  </tr>
  <tr>
    <td colspan="2">
      <button (click)="postBook()">Submit</button>
    </td>
  </tr>
```

```
</table>
<table border="1">
  <tr>
    <td>BookId</td>
    <td>BookName</td>
    <td>Price</td>
    <td>Image</td>
  </tr>
  <tbody>
    <tr *ngFor="let book of books">
      <td>{{book.BookId}}</td>
      <td>{{book.BookName}}</td>
      <td>{{book.Price}}</td>
      <td>
        
      </td>
    </tr>
  </tbody>
</table>
```

Table thứ nhất để cho nhập liệu thông tin chi tiết của Book, dùng [(ngModel)] để binding.

Table thứ hai để hiển thị danh sách Book lấy từ server. Nếu tái sử dụng component thì ta có thể thay thế table này thành 1 lệnh duy nhất:

```
<div app-books></div>
```

Cấu hình routing hoặc tham chiếu trực tiếp vào “app.component.html” để chạy component này ta sẽ có kết quả như đề bài yêu cầu.

Bài 119 – Tao HTTP PUT – Update a Book

Yêu cầu:

Viết Restful API HTTP PUT để cập nhật thông tin của 1 Book khi biết BookId.

Hướng dẫn:

Bước 1: Mở file **index.js** trong project “**my-server**”, bổ sung Restful API cập nhật book như dưới đây:

```
app.put("/books", cors(), (req, res) => {
    book = database.find(x => x.BookId == req.body.BookId)
    if(book != null)
    {
        book.BookName = req.body.BookName
        book.Price = req.body.Price
        book.Image = req.body.Image
    }
    res.send(database)
})
```

Bước 2: Thủ nghiệm Postman, để đảm bảo rằng API chạy được trước khi gắn vào Front End để lập trình:

The screenshot shows the Postman interface with the following details:

- Request Method:** PUT
- Request URL:** http://localhost:3000/books
- Body (JSON):**

```

1 "BookId": "b6",
2   "BookName": "Data Mining",
3   "Price": 555,
4   "Image": "b66.png"
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38 ]
```
- Response:** 200 OK, 3 ms, 773 B

- Chọn method là PUT
- Chọn raw/JSON
- Thẻ body nhập cấu trúc Book Json để gửi lên Server

```
{
  "BookId": "b6",
  "BookName": "Data Mining",
  "Price": 555,
  "Image": "b66.png"
}
```

-Nhấn nút “send” ta thấy Server đã chỉnh sửa dữ liệu thành công khi trả về danh sách dữ liệu mới nhất với các thông tin vừa được cập nhật.

Bài 120 – Triệu gọi HTTP PUT – Update a Book**Yêu cầu:**

Thiết kế giao diện tương tự như triệu gọi HTTP POST, tuy nhiên trường hợp này là gửi dữ liệu lên để chỉnh sửa:

Book Id:	Nhập Book Id		
Book Name:	Nhập Book Name		
Price:	0		
<input type="button" value="Update"/>			
BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	
b6	Data Mining	555	

Hướng dẫn:**Bước 1:**

Mở file “book-api.service.ts” để bổ sung hàm putBook cho lớp BookAPIService.

```
putBook(aBook:any):Observable<any>
{
  const headers=new HttpHeaders().set("Content-Type","application/json;charset=utf-8")
  const requestOptions:Object={
    headers:headers,
    responseType:"text"
  }
  return
  this._http.put<any>("/books",JSON.stringify(aBook),requestOptions).pipe(
    map(res=>JSON.parse(res) as Array<IBook>),
    retry(3),
    catchError(this.handleError))
}
```

Bước 2:

Tạo thêm một component **BookUpdate**.

File “**book-update.component.ts**” chỉnh mã lệnh:

```
export class BookUpdateComponent {  
  book=new Book();  
  books:any;  
  errorMessage:string='';  
  constructor(private _service: BookAPIService){  
    this._service.getBooks().subscribe({  
      next:(data)=>{this.books=data},  
      error:(err)=>{this.errorMessage=err}  
    })  
  }  
  putBook()  
{  
  this._service.putBook(this.book).subscribe({  
    next:(data)=>{this.books=data},  
    error:(err)=>{this.errorMessage=err}  
  })  
}  
}  
}
```

File “**book-update.component.html**” chỉnh mã lệnh:

```
<p>book-update works!</p>  
<p>{{errorMessage}}</p>  
<table>  
  <tr>  
    <td>Book Id:</td>  
    <td>  
      <input type="text" class="form-control" #name="ngModel" name="BookId"  
      id="BookId" placeholder="Nhập Book Id"  
      [(ngModel)]="book.BookId"/>  
    </td>  
  </tr>  
  <tr>  
    <td>Book Name:</td>  
    <td>  
      <input type="text" class="form-control" #name="ngModel"  
      name="BookName" id="BookName" placeholder="Nhập Book Name"  
      [(ngModel)]="book.BookName"/>  
    </td>  
  </tr>  
  <tr>  
    <td>Price:</td>  
    <td>  
      <input type="text" class="form-control" #name="ngModel"  
      name="BookPrice" id="BookPrice" placeholder="Nhập Price"  
      [(ngModel)]="book.Price"/>  
    </td>
```

```

</tr>
<tr>
    <td colspan="2">
        <button (click)="putBook()">Update</button>
    </td>
</tr>
</table>
<table border="1">
    <tr>
        <td>BookId</td>
        <td>BookName</td>
        <td>Price</td>
        <td>Image</td>
    </tr>
    <tbody>
        <tr *ngFor="let book of books">
            <td>{{book.BookId}}</td>
            <td>{{book.BookName}}</td>
            <td>{{book.Price}}</td>
            <td>
                
            </td>
        </tr>
    </tbody>
</table>

```

Cấu hình routing hoặc tham chiếu trực tiếp vào “app.component.html” để chạy component này ta sẽ có kết quả như đề bài yêu cầu.

Book Id:	b3																												
Book Name:	Introduction Machine Lea																												
Price:	350																												
<input type="button" value="Update"/>																													
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>BookId</th> <th>BookName</th> <th>Price</th> <th>Image</th> </tr> </thead> <tbody> <tr> <td>b1</td> <td>Kỹ thuật lập trình cơ bản</td> <td>70</td> <td></td> </tr> <tr> <td>b2</td> <td>Kỹ thuật lập trình nâng cao</td> <td>100</td> <td></td> </tr> <tr> <td>b3</td> <td>Máy học cơ bản</td> <td>200</td> <td></td> </tr> <tr> <td>b4</td> <td>Máy học nâng cao</td> <td>300</td> <td></td> </tr> <tr> <td>b5</td> <td>Lập trình Robot cơ bản</td> <td>250</td> <td></td> </tr> <tr> <td>b6</td> <td>Data Mining</td> <td>555</td> <td></td> </tr> </tbody> </table>		BookId	BookName	Price	Image	b1	Kỹ thuật lập trình cơ bản	70		b2	Kỹ thuật lập trình nâng cao	100		b3	Máy học cơ bản	200		b4	Máy học nâng cao	300		b5	Lập trình Robot cơ bản	250		b6	Data Mining	555	
BookId	BookName	Price	Image																										
b1	Kỹ thuật lập trình cơ bản	70																											
b2	Kỹ thuật lập trình nâng cao	100																											
b3	Máy học cơ bản	200																											
b4	Máy học nâng cao	300																											
b5	Lập trình Robot cơ bản	250																											
b6	Data Mining	555																											

Trước khi sửa

Book Id:	b3																												
Book Name:	Introduction Machine Lea																												
Price:	350																												
<input type="button" value="Update"/>																													
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>BookId</th> <th>BookName</th> <th>Price</th> <th>Image</th> </tr> </thead> <tbody> <tr> <td>b1</td> <td>Kỹ thuật lập trình cơ bản</td> <td>70</td> <td></td> </tr> <tr> <td>b2</td> <td>Kỹ thuật lập trình nâng cao</td> <td>100</td> <td></td> </tr> <tr> <td>b3</td> <td>Introduction Machine Learning</td> <td>350</td> <td></td> </tr> <tr> <td>b4</td> <td>Máy học nâng cao</td> <td>300</td> <td></td> </tr> <tr> <td>b5</td> <td>Lập trình Robot cơ bản</td> <td>250</td> <td></td> </tr> <tr> <td>b6</td> <td>Data Mining</td> <td>555</td> <td></td> </tr> </tbody> </table>		BookId	BookName	Price	Image	b1	Kỹ thuật lập trình cơ bản	70		b2	Kỹ thuật lập trình nâng cao	100		b3	Introduction Machine Learning	350		b4	Máy học nâng cao	300		b5	Lập trình Robot cơ bản	250		b6	Data Mining	555	
BookId	BookName	Price	Image																										
b1	Kỹ thuật lập trình cơ bản	70																											
b2	Kỹ thuật lập trình nâng cao	100																											
b3	Introduction Machine Learning	350																											
b4	Máy học nâng cao	300																											
b5	Lập trình Robot cơ bản	250																											
b6	Data Mining	555																											

Sau khi sửa

Bài 121– Tao HTTP DELETE– Remove a Book

Yêu cầu:

Viết Restful API để xóa 1 Book.

Hướng dẫn:

Trong project “my-server”, chỉnh sửa **index.js**:

```
app.delete("/books/:id", cors(), (req, res)=>{
    id=req.params["id"]
    database = database.filter(x => x.BookId !== id);
    res.send(database)
})
```

Trong Postman, method chọn DELETE, và url = “<http://localhost:3000/books/b5>”
Sau đó nhấn nút Send, chương trình sẽ xóa thành công Book có BookId=b5.

The screenshot shows the Postman interface. In the top right, there's a search bar with "Search Postman" and a "Sign In" button. Below it, the URL "http://localhost:3000/books/b5" is entered. The method dropdown shows "DELETE". The "Body" tab is selected, and the content type is set to "JSON". The body field contains the following JSON:

```
10 |     "BookName": "Kỹ thuật lập trình nâng cao",
11 |     "Price": 100,
12 |     "Image": "b2.png"
13 | },
14 | {
15 |     "BookId": "b3",
16 |     "BookName": "Máy học cơ bản",
17 |     "Price": 200,
18 |     "Image": "b3.png"
19 | },
20 | {
21 |     "BookId": "b4",
22 |     "BookName": "Máy học nâng cao",
23 |     "Price": 300,
24 |     "Image": "b4.png"
25 | },
26 | ]
```

At the bottom of the interface, there are buttons for "Find and Replace", "Console", "Runner", and "Trash".

Bài 122 – Triệu gọi HTTP DELETE– Remove a Book

Yêu cầu:

Thiết kế giao diện để xóa Book bằng cách triệu gọi HTTP Delete API.

Book Id: b5			Delete Book
BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	

-Mặc định danh sách Book sẽ hiển thị lên giao diện

-Nhập Book Id, nhấn nút Delete Book để xóa đồng thời cập nhật lại danh sách nếu xóa thành công

Hướng dẫn:

Bước 1: Mở file “book-api.service.ts” và Bổ sung mã lệnh cho **BookAPIService**:

```
deleteBook(bookId:string):Observable<any>
{
  const headers=new HttpHeaders().set("Content-Type","application/json;charset=utf-8")
  const requestOptions:Object={
    headers:headers,
    responseType:"text"
  }
  return this._http.delete<any>("/books/"+bookId,requestOptions).pipe(
    map(res=>JSON.parse(res) as Array<IBook>),
    retry(3),
    catchError(this.handleError)
}
```

Bước 2: Thêm component BookDelete.

Bổ sung mã lệnh cho “book-delete.component.ts”:

```
import { Component } from '@angular/core';
import { BookAPIService } from '../book-api.service';

@Component({
```

```
selector: 'app-book-delete',
templateUrl: './book-delete.component.html',
styleUrls: ['./book-delete.component.css']
})
export class BookDeleteComponent {
books:any
errMessage:string=''
constructor(private _service: BookAPIService){
  this._service.getBooks().subscribe({
    next:(data)=>{this.books=data},
    error:(err)=>{this.errMessage=err}
  })
}
deleteBook(bookId:any)
{
  this._service.deleteBook(bookId).subscribe({
    next:(data)=>{this.books=data},
    error:(err)=>{this.errMessage=err}
  })
}
}
```

Bổ sung mã lệnh cho “book-delete.component.html”:

```
<p>book-delete works!</p>
<p>{{errMessage}}</p>
Book Id: <input type="text" #bookId> <button
(click)="deleteBook(bookId.value)">Delete Book</button>
<table border="1">
  <tr>
    <td>BookId</td>
    <td>BookName</td>
    <td>Price</td>
    <td>Image</td>
  </tr>
  <tbody>
    <tr *ngFor="let book of books">
      <td>{{book.BookId}}</td>
      <td>{{book.BookName}}</td>
      <td>{{book.Price}} </td>
      <td>
        
      </td>
    </tr>
  </tbody>
</table>
```

Cấu hình routing hoặc tham chiếu trực tiếp vào “app.component.html” để chạy component này ta sẽ có kết quả như đề bài yêu cầu.

Book Id: b5		Delete Book	
BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	
b5	Lập trình Robot cơ bản	250	

Trước khi xóa

Book Id: b5		Delete Book	
BookId	BookName	Price	Image
b1	Kỹ thuật lập trình cơ bản	70	
b2	Kỹ thuật lập trình nâng cao	100	
b3	Máy học cơ bản	200	
b4	Máy học nâng cao	300	

Sau khi xóa

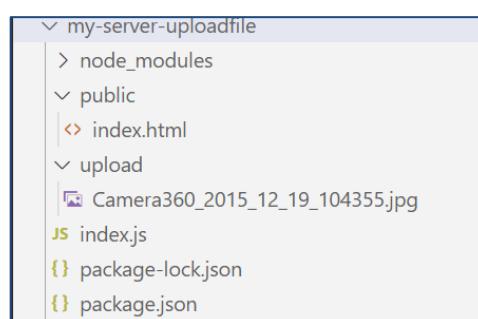
Bài 123 – Upload files to Server-1

Yêu cầu:

Tạo một Project NodeJs mới mang tên “my-server-uploadfile”, viết lệnh để upload file lên Server. Trường hợp này màn hình load file nằm trong Server

Hướng dẫn:

Bước 1: Tạo Project “my-server-uploadfile” có cấu trúc như hình bên dưới:



-Tạo các thư mục public, upload

Bước 2: Cài đặt các lệnh:

```
npm install express
npm install express-fileupload
npm i --save-dev nodemon
npm i --save-dev morgan
npm i body-parser
```

-Sau khi cài đặt xong và cấu hình “package.json”

```
{  
  "name": "my-server-uploadfile",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "start": "nodemon index.js",  
    "test": "echo \\"$Error: no test specified\\" && exit 1"  
  },  
  "author": "Tran Duy Thanh",  
  "license": "ISC",  
  "dependencies": {  
    "express": "^4.18.2",  
    "express-fileupload": "^1.4.0"  
  },  
  "devDependencies": {  
    "morgan": "^1.10.0",  
    "nodemon": "^2.0.21"  
  }  
}
```

Bước 3: Viết mã lệnh cho index.js:

```
const express = require('express');  
const fileUpload = require('express-fileupload');  
const app = express();  
const port = 3001;  
const morgan=require("morgan")  
app.use(morgan("combined"))  
const bodyParser=require("body-parser")  
app.use(bodyParser.json());  
app.use(bodyParser.urlencoded({extended: true}));  
  
app.use(  
  fileUpload({  
    limits: {  
      fileSize: 10000000,  
    },  
    abortOnLimit: true,  
  })  
);  
  
// Add this line to serve our index.html page  
app.use(express.static('public'));  
  
app.get('/', (req, res) => {  
  res.sendFile('index.html');  
});  
app.post('/upload', (req, res) => {  
  // Get the file that was set to our field named "image"  
  const { image } = req.files;
```

```
// If no image submitted, exit
if (!image) return res.sendStatus(400);

// Move the uploaded image to our upload folder
image.mv(__dirname + '/upload/' + image.name);

// All good
res.sendStatus(200);
});

app.listen(port, () => {
  console.log(`Example app listening on port ${port}`);
});
```

-Bước 4: Tạo index.html trong thư mục public

```
<!DOCTYPE html>
<form action="/upload" method="POST" enctype="multipart/form-data">
  <input type="file" name="image" />
  <button type="submit">Upload</button>
</form>
```

-Chạy ứng dụng Server: npm start

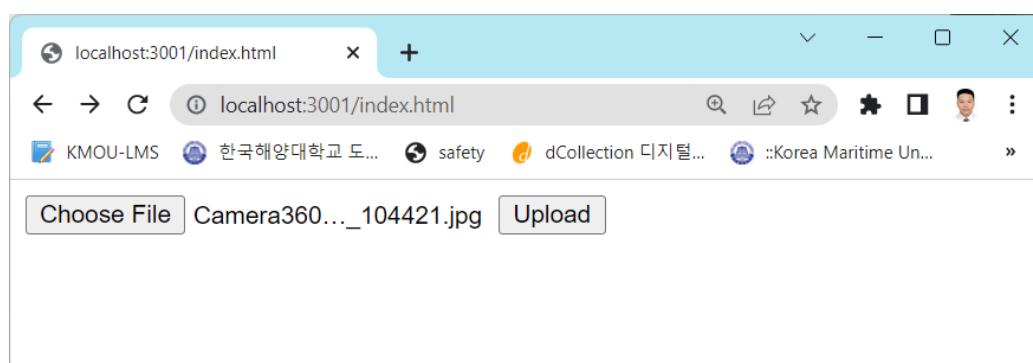
```
PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server-uploadfile> npm start
```

```
> my-server-uploadfile@1.0.0 start
> nodemon index.js

[nodemon] 2.0.21
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Example app listening on port 3001
```

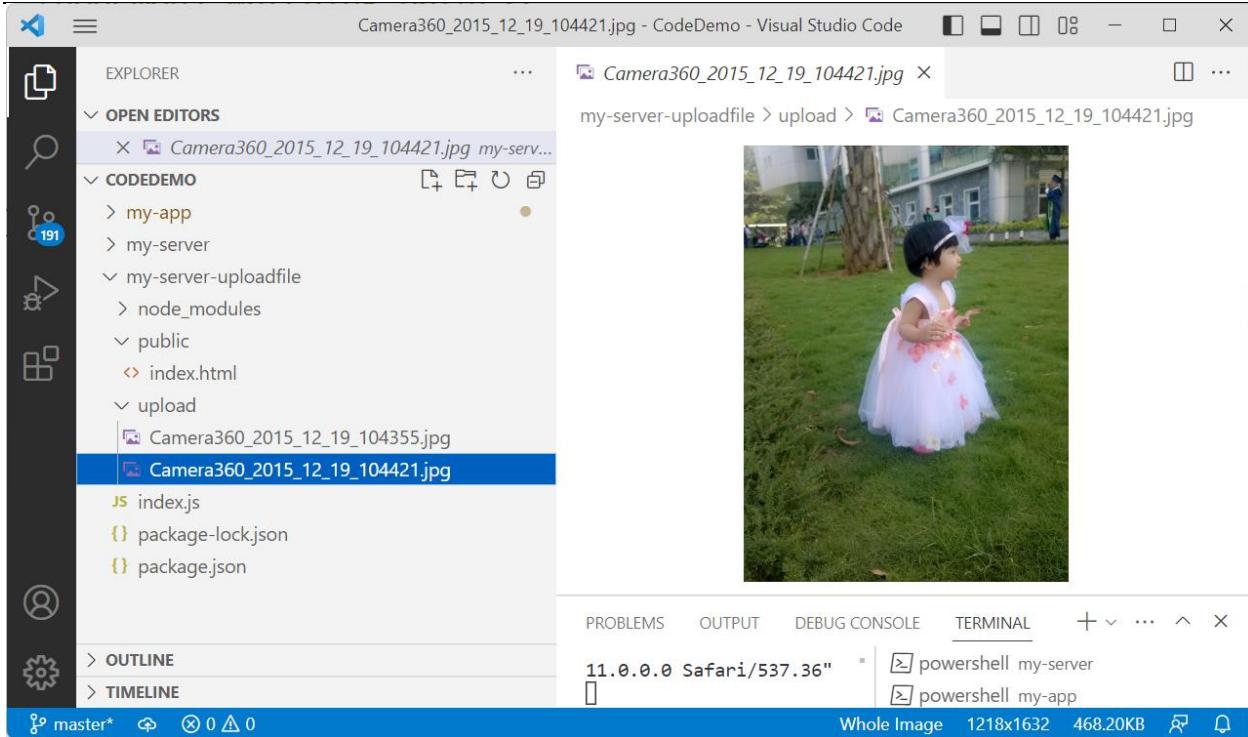
Chạy trình duyệt:

<http://localhost:3001/index.html>



Bấm Choose File, chọn hình rồi bấm upload

Kết quả sau khi upload thành công:



Bài 124 – Upload files to Server-2

Yêu cầu:

Tái sử dụng “my-server-uploadfile” xem như là 1 Service độc lập dùng để upload hình ảnh(và ta không cần phải lập trình lại hàm upload), hoặc ta có thể di chuyển các lệnh trong “index.js” của project “my-server-uploadfile” sang “index.js” của project “my-server”.

Sinh viên áp dụng bài này để cập nhật hình ảnh cho Books ở các bài trước.

Khi upload được hình ảnh lên Server rồi thì ta đã có đường dẫn vật lý lưu trữ trên Server (mà không phải trong thư mục assets), khi nạp dữ liệu Json trả về cho HTML hiển thị hình ảnh thì ta chỉ cần sửa lại giá trị thuộc tính “Image” của đối tượng trả cho đúng đường dẫn của hình ảnh đã đưa lên server là được.

Hướng dẫn:

Trong bài này, tài liệu sẽ hướng dẫn tái sử dụng “my-server-uploadfile” và trình thêm cách xem lại hình ảnh đã tải lên. Dựa vào bài này Sinh viên có thể dễ dàng áp dụng cho phần cập nhật hình ảnh của Books ở các bài trước.

Bước 1: Cài đặt thêm thư viện cors cho “my-server-uploadfile”

npm i cors

Bước 2: Mở file index.js của project “my-server-uploadfile” và bổ sung thêm API Get để tải hình ảnh từ server về client khi biết tên hình nhầm hiển thị nó lên HTML

The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with project files like 'file-upload.component.html', 'file-upload.component.ts', 'index.js', 'my-server-uploadfile', 'node_modules', 'public', 'upload', and 'package-lock.json'. The main editor area shows 'index.js' with the following code:

```

21 app.use(express.static('public'));
22
23 app.get('/', (req, res) => {
24   res.sendFile('index.html');
25 });
26
27 const cors=require("cors")
28 app.use(cors())
29 app.get("/image/:id",cors(),(req,res)=>{
30   id=req.params["id"]
31   console.log('upload/'+id)
32   res.sendFile(__dirname+'/upload/'+id);
33 })
34 app.post('/upload', (req, res) => {
35

```

A yellow highlight covers the CORS configuration and the GET handler for images.

Ví dụ, khi có 1 tên hình nào đó thì ở client (AngularJS) ở các component ta sẽ gọi:

```
<img src='http://localhost:3001/image/metaverse.png'>
```

Nó sẽ tự động gọi API GET để hiển thị chi tiết hình ảnh lên giao diện HTML

Tức là nếu áp dụng để lưu hình ảnh cho **book** thì ta chỉ cần lưu tên hình khi tạo một Book mới lên Server. Ở ví dụ này thì **book.image**= “metaverse.png”, lúc này

Thay vì các bài trước viết là:

```

```

Thì chuyển thành:

```

```

Dưới đây là mã lệnh đầy đủ của index.js trong server “my-server-uploadfile”, lưu ý Webserver này ta chạy ở port 3001

```

const express = require('express');
const fileUpload = require('express-fileupload');
const app = express();
const port = 3001;
const morgan=require("morgan")
app.use(morgan("combined"))
const bodyParser=require("body-parser")
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

app.use(
  fileUpload({
    limits: {
      fileSize: 10000000,
    },
    abortOnLimit: true,
  })
);

// Add this line to serve our index.html page
app.use(express.static('public'));

```

```
app.get('/', (req, res) => {
    res.sendFile('index.html');
});

const cors=require("cors")
app.use(cors())
app.get("/image/:id",cors(),(req,res)=>{
    id=req.params["id"]
    console.log('upload/'+id)
    res.sendFile(__dirname+ '/upload/' +id);
})
app.post('/upload', (req, res) => {

    // Get the file that was set to our field named "image"
    const { image } = req.files;

    // If no image submitted, exit
    if (!image) return res.sendStatus(400);

    // Move the uploaded image to our upload folder
    image.mv(__dirname + '/upload/' + image.name);

    // All good
    res.sendStatus(200);
});

app.listen(port, () => {
    console.log(`Example app listening on port ${port}`);
});
```

Bước 3: Mở dự án “my-app” để tham chiếu sử dụng Restful API trong “my-server-uploadfile”, như vậy “my-app” đã sử dụng nhiều Restful API ở các Web server khác nhau (nó cũng là đúng với thực tế).

-Mở file “proxy.conf.json” bổ sung khai báo:

```
{
    "context": ["/upload"],
    "target": "http://localhost:3001",
    "secure": true,
    "changeOrigin":true,
    "logLevel": "debug"
}
```

Khai báo đầy đủ của “proxy.conf.json”:

```
[{
    {
        "context": ["/exchange"],
        "target": "https://www.dongabank.com.vn",
        "secure": true,
```

```
        "changeOrigin":true,
        "logLevel":"debug"
    },
    {
        "context": ["/products"],
        "target": "https://fakestoreapi.com",
        "secure": true,
        "changeOrigin":true,
        "logLevel":"debug"
    },
    {
        "context": ["/books"],
        "target": "http://localhost:3000",
        "secure": true,
        "changeOrigin":true,
        "logLevel":"debug"
    },
    {
        "context": ["/upload"],
        "target": "http://localhost:3001",
        "secure": true,
        "changeOrigin":true,
        "logLevel":"debug"
    }
]
```

Bước 4: Tạo thêm một component mới với tên “FileUpload”.

-Mã lệnh “file-upload.component.html” như sau:

```
<input type="file" class="file-input"
       (change)="onFileSelected($event) #fileUpload name="image">

<div class="file-upload">
    {{fileName || "No file uploaded yet."}}
    <button mat-mini-fab color="primary" class="upload-btn"
           (click)="fileUpload.click()">
        Upload
    </button>
</div>
```

-Mã lệnh của “file-upload.component.ts” như sau:

```
import { HttpClient, HttpEventType } from '@angular/common/http';
import { Component, Input } from '@angular/core';
import { finalize, Subscription } from 'rxjs';

@Component({
    selector: 'app-file-upload',
    templateUrl: './file-upload.component.html',
```

```
        styleUrls: ['./file-upload.component.css']
    })
export class FileUploadComponent {
    @Input()
    requiredFileType:any;

    fileName = '';
    uploadProgress:number=0;
    uploadSub: Subscription=new Subscription();

    constructor(private http: HttpClient) {}

    onFileSelected(event:any) {
        const file:File = event.target.files[0];

        if (file) {
            this.fileName = file.name;
            const formData = new FormData();
            formData.append("image", file);

            const upload$ = this.http.post("/upload", formData, {
                reportProgress: true,
                observe: 'events'
            })
            .pipe(
                finalize(() => this.reset())
            );

            this.uploadSub = upload$.subscribe(event => {
                if (event.type == HttpEventType.UploadProgress) {
                    this.uploadProgress = Math.round(100 * (event.loaded /
event.total));
                }
            })
        }
    }

    cancelUpload() {
        this.uploadSub.unsubscribe();
        this.reset();
    }

    reset() {
        this.uploadProgress = 0;
        this.uploadSub = new Subscription();
    }
}
```

Cấu hình Routing hoặc tham chiếu trực tiếp “app.component.html” để chạy ta sẽ có kết quả như mong muốn. Sinh viên suy nghĩ để tích hợp phần hình ảnh này vào cho Book.

Bài 125 – Bài tập tổng hợp Restful API(*)

Yêu cầu:

- Xây dựng Restful API cung cấp các chức năng: lấy toàn bộ thông tin sách, xem chi tiết sách, tạo sách mới, chỉnh sửa sách, xóa sách.
- Thiết kế màn hình quản lý thông tin sách như hình minh họa:

QUẢN LÝ THÔNG TIN SÁCH

Create New

Tensach	Giaban	Mota	Anhbia	Ngaycapnhat	Soluongton	MaCD	MaNXB	
Giáo trình Tin học cơ bản	26000.00	Nội dung của cuốn: Tin Học Cơ Bản Windows XP gồm có 7 chương: Chương 1: Một số vấn đề cơ bản. Chương 2: Sử dụng nhanh thanh công cụ và thanh thực đơn trong My Computer và Windows Explorer. Chương 3: Các thao tác trong windows XP. Chương 4: Các thiết lập trong Windows XP. Chương 5: Bảo trì máy tính. Chương 6: Các phím tắt Chương 7: Hồi và đập các thắc mắc. Xin trân trọng giới thiệu cuốn sách cùng bạn	THCB.jpg	25/10/2014 12:00:00 SA	120	7	1	Edit Details Delete
Giáo trình Cơ Sở Dữ Liệu Với Visual Basic 2005 Và ADO.NET 2.0	12000.00	Cuốn sách này gồm 3 phần sau: Phần 1: Xử lý văn bản trong Microsoft Word. Phần 2: Quản lý cơ sở dữ liệu. Phần 3: Tạo báo cáo với Crystal Report. Cuốn sách này gồm 3 phần sau: Phần 1: Xử lý văn bản trong Microsoft Word. Phần 2: Quản lý cơ sở dữ liệu. Phần 3: Tạo báo cáo với Crystal Report.	TH004.jpg	23/10/2013 12:00:00 SA	25	3	2	Edit Details Delete
Visual Basic 2005 Tập 3, Quyển 2: Lập Trình Web Với Cơ Sở Dữ Liệu	20000.00	"Visual Basic 2005 Tập 3, Quyển 2: Lập Trình Web Với Cơ Sở Dữ Liệu" sẽ cung cấp kỹ thuật và hướng dẫn ban đầu về cách thiết kế và phát triển ứng dụng Web. Cuốn sách bao gồm các bài học và bài tập minh họa, giúp bạn nắm vững cách sử dụng ASP.NET để phát triển ứng dụng Web.	LTWeb2005.jpg	15/09/2014 12:00:00 SA	240	8	4	Edit Details Delete

- +Mặc định chương trình sẽ gọi API GET all Books để hiển thị.
- +Nhấn nút “Create New” hiển thị màn hình thêm sách, khi thêm thành công sẽ quay lại màn hình danh sách ban đầu
- +Nhấn nút “Edit” hiển thị màn hình chỉnh sửa thông tin sách, chỉnh sửa thành công sẽ quay lại màn hình danh sách ban đầu
- + Nhấn nút “Details” hiển thị thông tin chi tiết của sách đang chọn
- + Nhấn nút “Delete” sẽ xác nhận hỏi xóa hay không, nếu xóa thành công thì nạp lại danh sách sau khi xóa.
- Sinh viên tự thiết kế cấu trúc JSon cho Book để đáp ứng yêu cầu đề bài.
- Xử lý hình ảnh cho Book.

Module 12 –Restful API với MongoDB

Nội dung kiến thức thực hành:

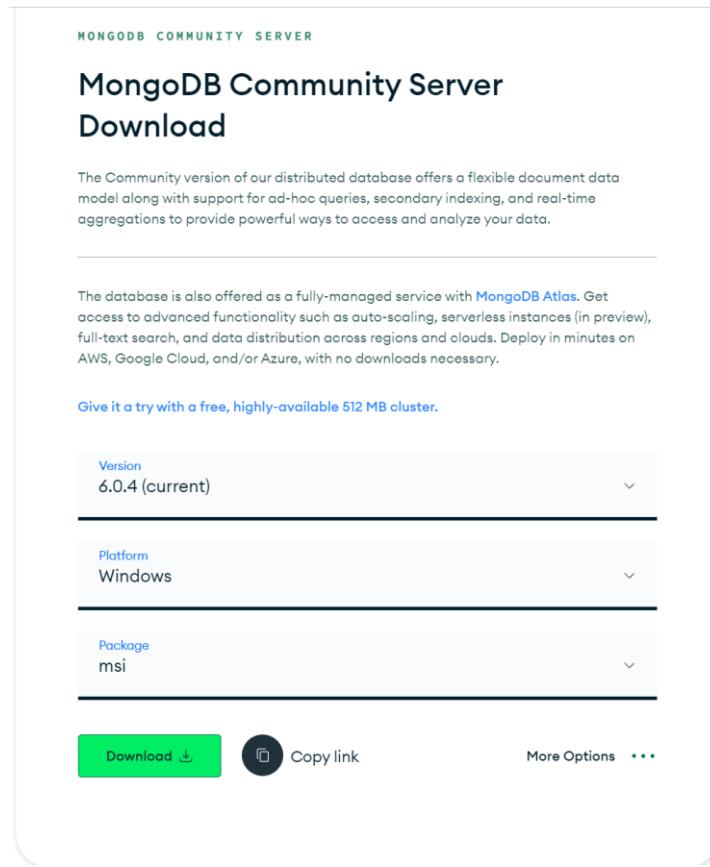
- + Cung cấp các kiến thức và kỹ năng lập trình Restful API
- + Sử dụng MongoDB và MongoDB Compass
- + Các thư viện kết nối MongoDB
- + Tạo Restful API với MongoDB
- + Lưu trữ hình ảnh trong MongoDB

Bài 126- Cài đặt và sử dụng MongoDB

Yêu cầu:

Để tải MongoDB, ta vào link

<https://www.mongodb.com/try/download/community>



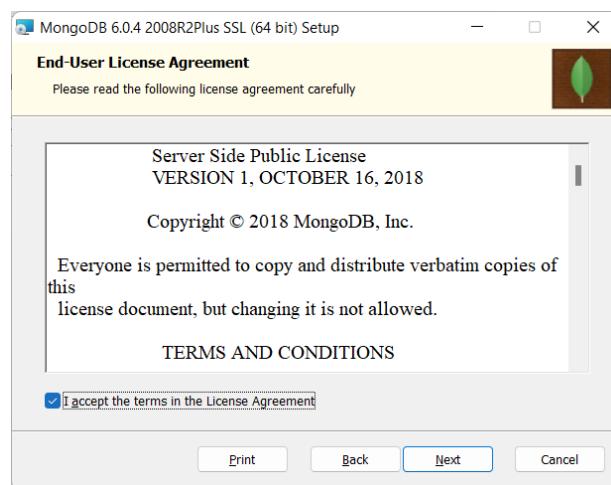
The screenshot shows the MongoDB Community Server Download page. At the top, it says "MONGODB COMMUNITY SERVER". Below that is the heading "MongoDB Community Server Download". A descriptive text block follows, stating: "The Community version of our distributed database offers a flexible document data model along with support for ad-hoc queries, secondary indexing, and real-time aggregations to provide powerful ways to access and analyze your data." Another text block below it says: "The database is also offered as a fully-managed service with MongoDB Atlas. Get access to advanced functionality such as auto-scaling, serverless instances (in preview), full-text search, and data distribution across regions and clouds. Deploy in minutes on AWS, Google Cloud, and/or Azure, with no downloads necessary." A "Give it a try with a free, highly-available 512 MB cluster." button is present. Below these are three dropdown menus: "Version" set to "6.0.4 (current)", "Platform" set to "Windows", and "Package" set to "msi". At the bottom are "Download" and "Copy link" buttons, and a "More Options" link.

Nhấn Download ta được file “mongodb-windows-x86_64-6.0.4-signed.msi”

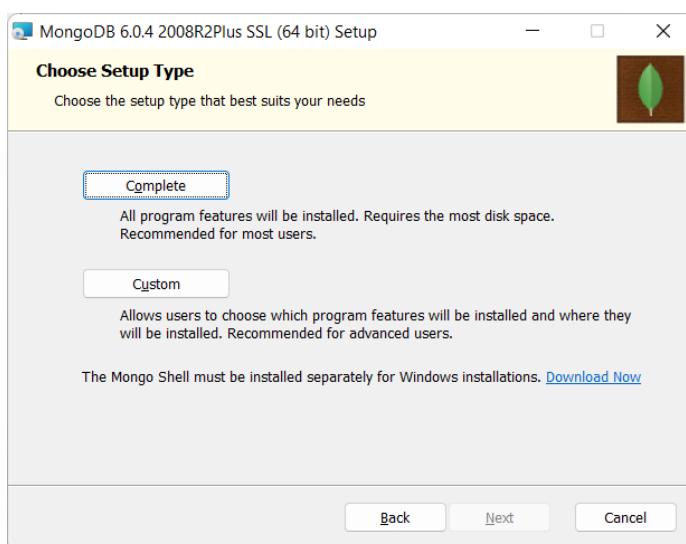
Bấm vào file để cài đặt



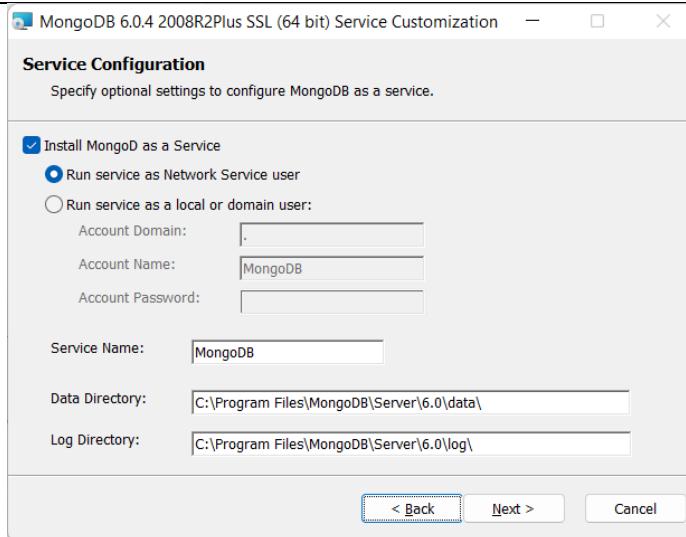
Nhấn Next



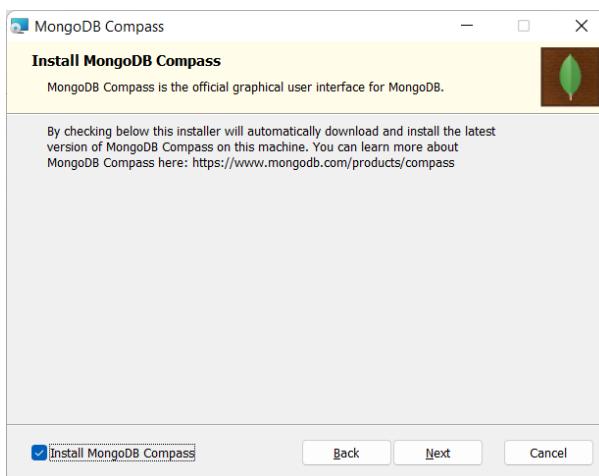
Checked “I accept the terms in the License Agreement” và nhấn Next



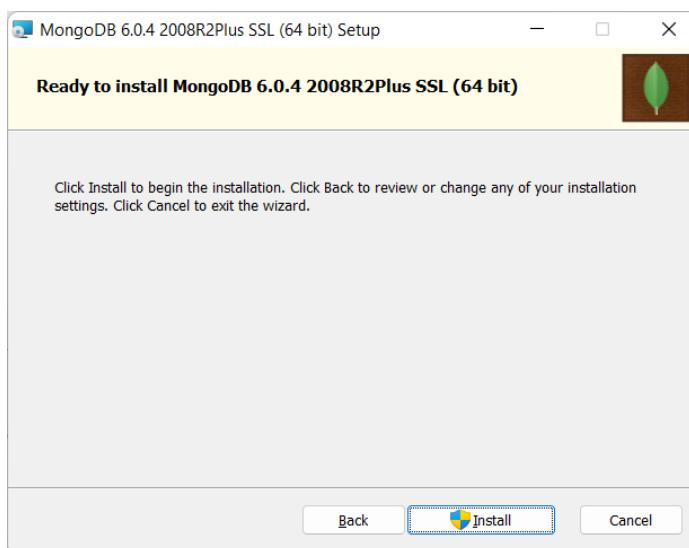
Nhấn nút complete



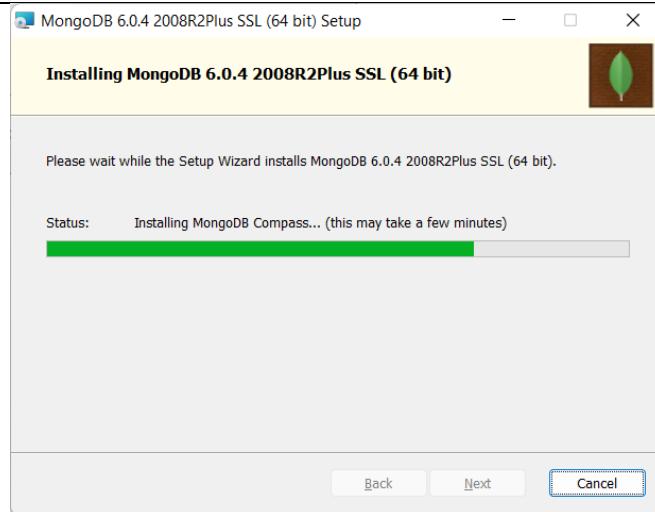
Để mặc định như hình trên và bấm Next



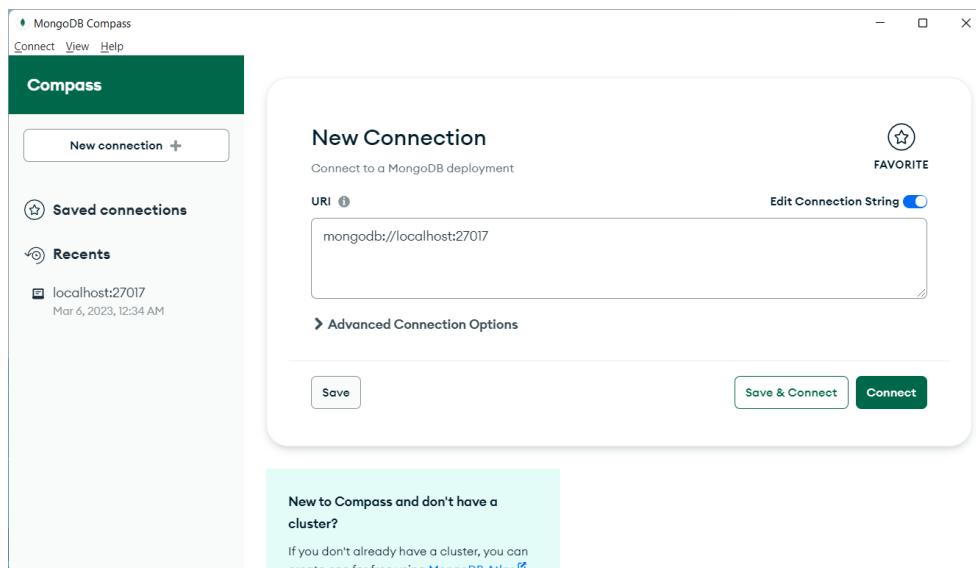
Checked “Install MongoDB Compass” rồi bấm Next. Compass là công cụ cho ta thao tác với MongoDB Server



Bấm Install



Chờ hệ thống cài đặt cho tới lúc thành công. Đây là màn hình MongoDB Compass khi cài đặt xong.

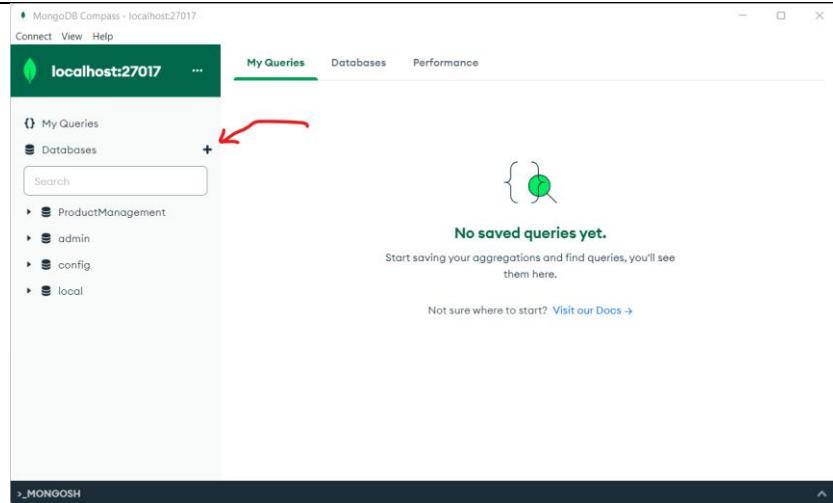


Đường dẫn kết nối:

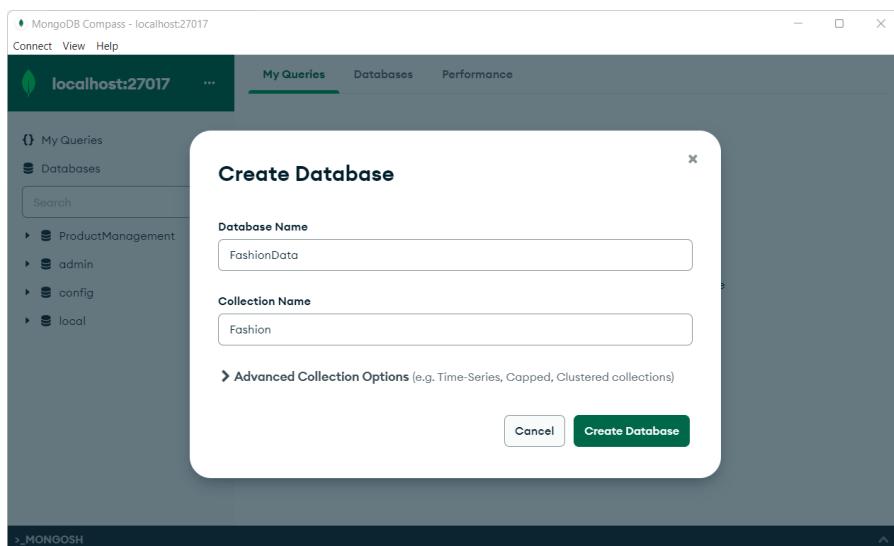
mongodb://localhost:27017

Đôi khi trong quá trình lập trình, có thể một số thư viện set mặc định IPv6 nếu để localhost. Như vậy máy tính sẽ không thể lập trình được nếu nó là IPv4. Lúc này ta nên thay localhost bằng 127.0.0.1

Bấm connect để kết nối:



Để tạo Cơ sở dữ liệu ta bấm vào biểu tượng dấu “+” kế bên chữ “Databases” và đặt tên cơ sở dữ liệu như hình:



Database Name: FashionData

Collection Nam: Fashion

SQL		NoSQL
Database	➔	Database
Table	➔	Collection
Row	➔	Document
Column	➔	Field/Attribute

MongoDB Compass - localhost:27017/FashionData.Fashion

Connect View Collection Help

localhost:27017 ...

Documents FashionData.Fas...

My Queries Databases Search

FashionData Fashion

Documents Aggregations Schema Explain Plan In...

Filter Type a query: { field: 'value' }

ADD DATA EXPORT COLLECTION

Import file Insert document

Open in new tab Drop collection

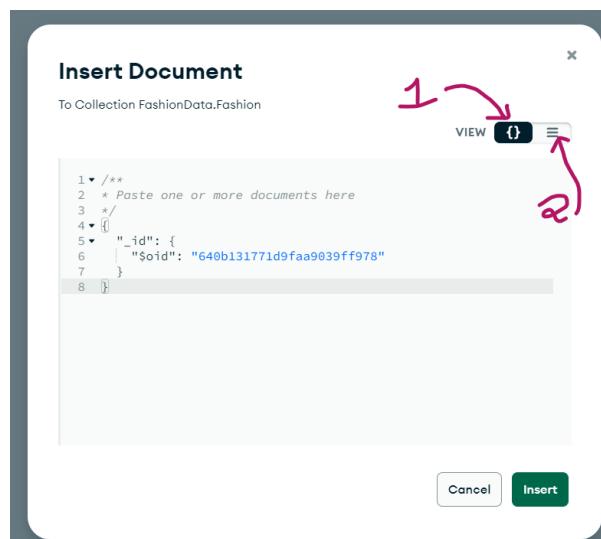
Mục số 1:

- “Open in new tab”: Mở thêm 1 Tab mới để làm việc
- “Drop collection”: Xóa collection đang chọn

Mục số 2:

- “Import File”: Nhập dữ liệu từ một file có sẵn trước đó
- “Insert document”: Thêm mới một dữ liệu, nó thường là 1 JSON (ta có thể tạo 1 cấu trúc phức tạp chứ không nhất thiết là 1 JSONObject)

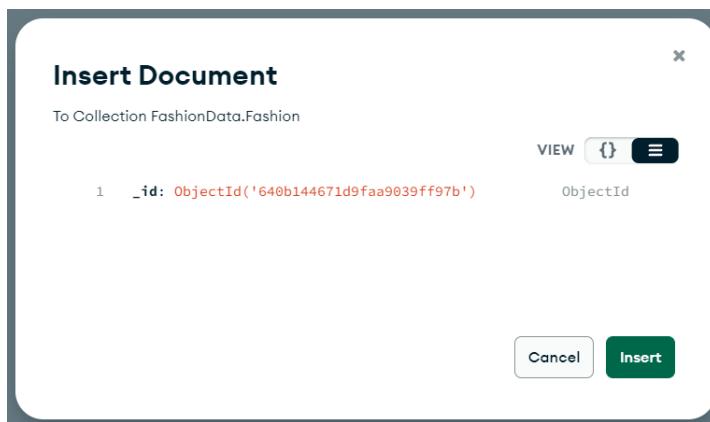
Bây giờ ta chọn “Insert document”:



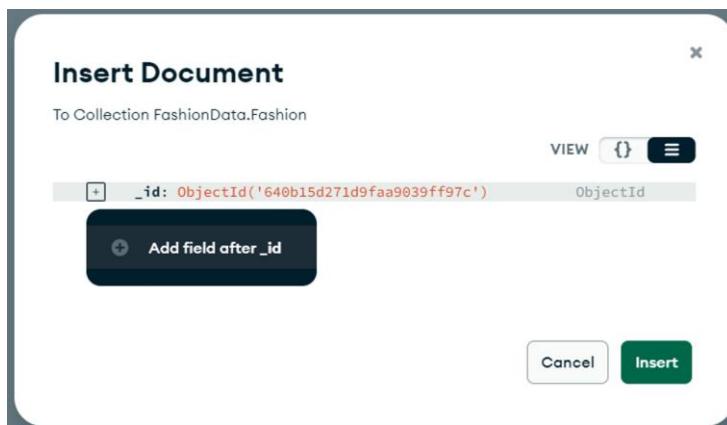
-Cách nhập theo số 1 là định dạng cấu trúc Json trực tiếp

-Cách nhập theo số 2 là định dạng Visualization dạng Field và cho lựa chọn kiểu dữ liệu của mỗi Field.

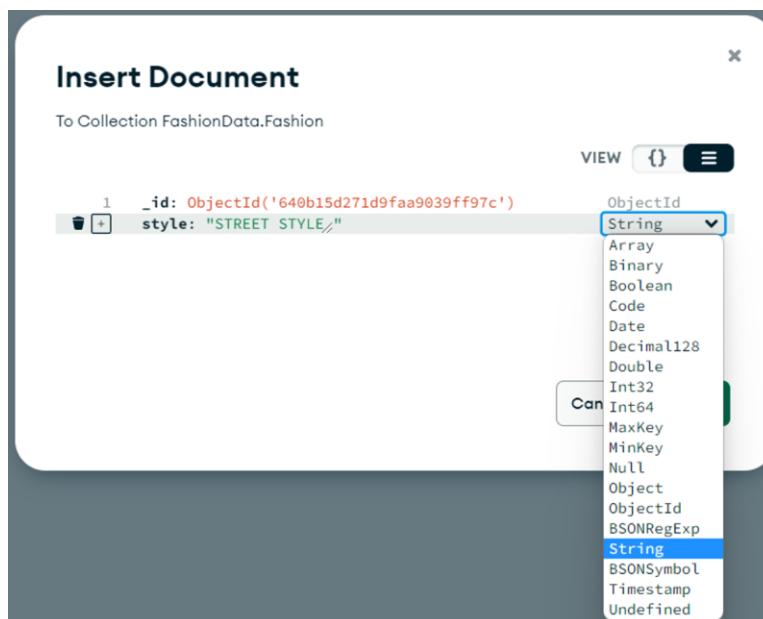
Ta lựa chọn cách số 2 để dễ thiết kế:



Di chuyển chuột tới đằng trước 1 Field bất kỳ, nó hiển thị ra dấu “+”, bấm chuột vào dấu “+” và chọn “Add field after _id” như hình dưới đây:



Ta tạo một Field tên là style và giá trị “STREET STYLE” và kiểu dữ liệu là String như hình minh họa dưới đây:



Tương tự như vậy, ta tiếp tục tạo thêm một số Field cho Collection Fashion này:



Hình ảnh là Base64 String. Dùng tool này để đưa hình về string rồi dán vào MongoDB: <https://codebeautify.org/image-to-base64-converter>

Đây là minh họa tạo cấu trúc và nhập document cho collection Fashion:

```
_id: ObjectId('640821071d09faa9039ff97e')
style: "STREET STYLE"
fashion_subject: "Mil Oh's Best Street Style Photos From the Fall 2023 Shows in Paris"
fashion_detail: "There are two street style camps in Paris this season-those who are w/."
fashion_image: "data:image/jpeg;base64,iVBORw0KGgoAAAANSUhEUgAAQAAQIDCAVAAABIEhIAAA_"

_id: ObjectId('640821071d09faa9039ff97d')
style: "STREET STYLE"
fashion_subject: "Mil Oh's Best Street Style Photos From the Fall 2023 Shows in Milan"
fashion_detail: "In Milan, thanks to Diesel's Glenn Martens, there's double the denim."
fashion_image: "data:image/jpeg;base64,iVBORw0KGgoAAAANSUhEUgAAQAAQIDCAVAAACXX3yuAAA_"

_id: ObjectId('640821071d09faa9039ff980')
style: "STREET STYLE"
fashion_subject: "Vivienne Westwood Is Remembered in London"
fashion_detail: "How can you encapsulate the life and legacy of Vivienne Westwood-a chi_."
fashion_image: "data:image/jpeg;base64,iVBORw0KGgoAAAANSUhEUgAAQAAQIDCAVAAADicqMAAA_"

_id: ObjectId('640821071d09faa9039ff981')
style: "TRENDS"
fashion_subject: "What The Short Suit Should Be Your Next Spring Investment"
fashion_detail: "Everyone always talks about the red off-the-shoulder dress in Pretty W_."
fashion_image: "data:image/jpeg;base64,iVBORw0KGgoAAAANSUhEUgAAQAAQIDCAVAAABQHt9AAA_"

_id: ObjectId('640821071d09faa9039ff982')
style: "TRENDS"
fashion_subject: "Is This The Trend Report of the Future? An AI Interprets the Fall 2023."
fashion_detail: "Late last year, I visited Outlier designer Willie Norris at her Brookl_."
fashion_image: "data:image/jpeg;base64,iVBORw0KGgoAAAANSUhEUgAAQAAQIDCAVAAAD97jEDAAA_"

_id: ObjectId('640821071d09faa9039ff983')
style: "TRENDS"
fashion_subject: "Men, Skirts Aren't That Scary—Promise!"
fashion_detail: "I'll never forget the first time I wore a skirt. It was back in Septem_."
fashion_image: "data:image/jpeg;base64,iVBORw0KGgoAAAANSUhEUgAAQAAQIDCAVAAABQHt9AAA_"
```

Sinh viên có thể tải dữ liệu được Export ở đây:

<https://tranduythanh.com/webmaterials/Fashion.json>

Sau đó dùng chức năng import collection để tiết kiệm thời gian, cách import:

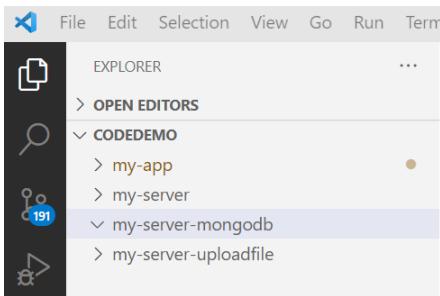
- Đầu tiên tạo 1 cơ sở dữ liệu tên “FashionData”

- Sau đó tạo 1 collection tên “Fashion”

- Cuối cùng chọn “Fashion” và import file, trả tới “Fashion.json”

Bài 127 - Cấu hình và Kết nối NodeJS với MongoDB**Yêu cầu:**

Tạo một thư mục đồng cấp “my-server-mongodb” để build Project cho Webserver tương tác MongoDB.

**Bước 1: Cài đặt các lệnh:**

```
npm install express
npm install express-fileupload
npm install mongodb
npm i --save-dev nodemon
npm i --save-dev morgan
npm i body-parser
npm i cors
```

Ngoài ra nếu dùng **mongoose** kết hợp thì ta có thể tải đặt thêm 2 thư viện:

```
npm i mongoose
npm install --save mongoclient
```

Trong bài này, tài liệu hướng dẫn dùng thư viện “**mongodb**”
File “**package.json**”:

```
{
  "name": "my-server-mongodb",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "nodemon index.js",
    "test": "echo \\"$Error: no test specified\\" && exit 1"
  },
  "author": "Tran Duy Thanh",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.20.2",
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "express-fileupload": "^1.4.0",
    "mongodb": "^5.1.0"
  }
},
```

```
"devDependencies": {  
    "morgan": "^1.10.0",  
    "nodemon": "^2.0.21"  
}  
}
```

Bước 2: Tạo file Index.js để viết API Service trong “my-server-mongodb”

Tạo file **index.js** cấu hình một số thư viện và một số hàm như đã học ở các bài trước, chạy ở Port 3002:

The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays a project structure under 'CODEDEMO'. It includes folders for 'my-app', 'my-server', and 'my-server-mongodb', which contains 'node_modules', 'index.js', 'package-lock.json', 'package.json', and 'my-server-uploadfile'. The 'index.js' file is currently open in the main editor area. The code in 'index.js' is as follows:

```
const express = require('express');  
const app = express();  
const port = 3002;  
  
const morgan=require("morgan")  
app.use(morgan("combined"))  
  
const bodyParser=require("body-parser")  
app.use(bodyParser.json());  
app.use(bodyParser.urlencoded({extended: true}));  
  
const cors=require("cors");  
app.use(cors());  
  
app.listen(port,()=>{
    console.log(`My Server listening on port ${port}`)
})  
  
app.get("/",(req,res)=>{
    res.send("This Web server is processed for MongoDB")
})  
  
const { MongoClient, ObjectId } = require('mongodb');
client = new MongoClient("mongodb://127.0.0.1:27017");
client.connect();
database = client.db("FashionData");
fashionCollection = database.collection("Fashion");
```

Các dòng lệnh từ số 1 tới 21 là cấu hình giống như đã học ở bài trước để xử lý các vấn đề liên quan tới Restful API.

Các dòng lệnh từ 23 tới 27 là để kết nối mongoDB database:

-Dòng 23 gọi thư viện **mongodb**

-Dòng 24 khai báo chuỗi kết nối tới Server, thay localhost bằng 127.0.0.1 nếu các thư viện lấy Ipv6 của máy tính

-Dòng 25 gọi lệnh kết nối tới Server

-Dòng 26 kết nối tới Database “FashionData” mà ta thiết kế ở bài trước.

-Dòng 27 truy suất collection Fashion. Nếu cơ sở dữ liệu có nhiều collection mà ta muốn truy suất thì cứ thêm các dòng khai báo mới.

Dưới đây là mã lệnh đầy đủ index.js:

```
const express = require('express');
const app = express();
const port = 3002;

const morgan=require("morgan")
app.use(morgan("combined"))

const bodyParser=require("body-parser")
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

const cors=require("cors");
app.use(cors())

app.listen(port,()=>{
    console.log(`My Server listening on port ${port}`)
})

app.get("/",(req,res)=>{
    res.send("This Web server is processed for MongoDB")
})

const { MongoClient, ObjectId } = require('mongodb');
client = new MongoClient("mongodb://127.0.0.1:27017");
client.connect();
database = client.db("FashionData");
fashionCollection = database.collection("Fashion");
```

Bước 3: Gọi lệnh **npm start** để chạy Service

The screenshot shows a terminal window with several tabs at the top: PROBLEMS (1), OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, displaying the following command and its output:

```
> my-server-mongodb@1.0.0 start
> nodemon index.js

[nodemon] 2.0.21
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
My Server listening on port 3002
□
```

Như vậy tới đây chương trình đã chạy ở port 3002 và kết nối thành công tới MongoDB

Bài 128 – Tao và triêu gọi API - HTTP GET – List Fashion**Yêu cầu:**

Tiếp tục cập nhật mã lệnh cho index.js trong project “my-server-mongodb”, khai báo API lấy toàn bộ Fashion và hiển thị danh sách Fashion này lên giao diện trong project “my-app”.

Hướng dẫn:

Bước 1: Vào “my-server-mongodb”, bổ sung hàm sau vào cuối file index.js (bổ sung đằng sau các lệnh kết nối tới MongoDB):

```
app.get("/fashions", cors(), async (req, res) =>{
  const result = await fashionCollection.find({}).toArray();
  res.send(result)
})
```

Lệnh trên truy vấn toàn bộ Fashion và trả về JSONArray. Vì ta đã cấu hình tự động restart server khi mã nguồn thay đổi nên chỉ cần mở Postman lên test chức năng này:

The screenshot shows the Postman application interface. In the top bar, there are tabs for Home, Workspaces, Explore, and a search field. Below the search field, there's a URL input field with the value "http://localhost:3002/fashions/" and a "Send" button. To the left, there's a sidebar with sections for Collections, APIs, Environments, Mock Servers, Monitors, and History. A central panel displays a collection named "fashions". Under this collection, there's a single GET request to "http://localhost:3002/fashions/". The "Body" tab is selected, showing a JSON response. The response is a single-element array with the following content:

```
1 [ { _id: "640b211171d9faa9039ff97f", style: "STREET STYLE", fashion_subject: "Vivienne Westwood Is Remembered in London", fashion_detail: "How can you encapsulate the life and legacy of Vivienne Westwood—a chief genius in England's fashion canon and much, much more besides—in a memorial service? As her granddaughter, Cora Corré, conceded in the last address of this afternoon's event at Southwark Cathedral: 'We can only really touch on the characteristics of the phenomenon that is Vivienne Westwood: a grandmother, a mother, a sister, a friend, a teacher, an artist, a designer—it will never be enough.'", fashion_image: "data:image/jpeg;base64, iVBORw0KGgoAAAANSUhEUgAAQYAAQD1icnMAAAAXNSR0TArs4c60AAAARn0II1RA" } ]
```

Ra được kết quả như trên là đã viết API fashions thành công.

Bước 2: Trong “my-app”, thêm mới 1 model class **Fashion** trong file “Fashion.ts”:

```
export class Fashion{  
    constructor(  
        public _id:any=null,  
        public style:string="",  
        public fashion_subject:string="",  
        public fashion_detail:string="",  
        public fashion_image:string=""){}  
}
```

Bước 3: Trong “my-app”, bổ sung Service FashionAPIService.

Viết mã lệnh cho “fashion-api.service.ts”:

```
import { HttpClient, HttpHeaders } from  
    '@angular/common/http';  
import { Injectable } from '@angular/core';  
import { catchError, map, Observable, retry, throwError } from 'rxjs';  
import { Fashion } from './models/Fashion';  
  
@Injectable({  
    providedIn: 'root'  
})  
export class FashionAPIService {  
  
    constructor(private _http: HttpClient) {}  
    getFashions():Observable<any>  
    {  
        const headers=new HttpHeaders().set("Content-Type", "text/plain;charset=utf-  
8")  
        const requestOptions:Object={  
            headers:headers,  
            responseType:"text"  
        }  
        return this._http.get<any>("/fashions",requestOptions).pipe(  
            map(res=>JSON.parse(res) as Array<Fashion>),  
            retry(3),  
            catchError(this.handleError))  
    }  
    handleError(error:HttpErrorResponse){  
        return throwError(()=>new Error(error.message))  
    }  
}
```

Bước 4: Trong “my-app”, bổ sung Componen FashionComponent.

Viết mã lệnh cho “fashion.component.ts”:

```
import { Component } from '@angular/core';
import { FashionAPIService } from '../fashion-api.service';

@Component({
  selector: 'app-fashion',
  templateUrl: './fashion.component.html',
  styleUrls: ['./fashion.component.css']
})
export class FashionComponent {
  fashions:any;
  errorMessage:string='';
  constructor(public _service: FashionAPIService){
    this._service.getFashions().subscribe({
      next:(data)=>{this.fashions=data},
      error:(err)=>{this.errorMessage=err}
    })
  }
}
```

Viết mã lệnh cho “fashion.component.html”:

```
<p>fashion works!</p>
{{errorMessage}}
<table border="1">
  <tr>
    <td>Id</td>
    <td>Fashion Subject</td>
    <td>Fashion Detail</td>
    <td>Image</td>
  </tr>
  <tbody>
    <tr *ngFor="let fashion of fashions">
      <td>{{fashion._id}}</td>
      <td>{{fashion.fashion_subject}}</td>
      <td>{{fashion.fashion_detail}}</td>
      <td>
        
      </td>
    </tr>
  </tbody>
</table>
```

Bước 5: Trong “my-app”, bổ sung khai báo cho “proxy.conf.json”:

```
{
  "context": ["/fashions"],
  "target": "http://localhost:3002",
  "secure": true,
  "changeOrigin":true,
  "logLevel": "debug"
}
```

Mã khai báo đầy đủ:

```
[  
  {  
    "context": ["/exchange"],  
    "target": "https://www.dongabank.com.vn",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
  },  
  {  
    "context": ["/products"],  
    "target": "https://fakestoreapi.com",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
  },  
  {  
    "context": ["/books"],  
    "target": "http://localhost:3000",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
  },  
  {  
    "context": ["/upload"],  
    "target": "http://localhost:3001",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
  },  
  {  
    "context": ["/fashions"],  
    "target": "http://localhost:3002",  
    "secure": true,  
    "changeOrigin":true,  
    "logLevel":debug  
  }  
,
```

Bước 6: Khai báo routing hoặc chạy component này trực tiếp trong “app.component.html”, ta có kết quả như mong muốn:

Id	Fashion Subject	Fashion Detail	Image
640b187e71d9faa9039ff97d	Phil Oh's Best Street Style Photos From the Fall 2023 Shows in Paris	<p>There are two street style camps in Paris this season—those who are willing to brave the cold and go coatless for the sake of fashion, and others who are bundling up in their warmest furs and scarves. Phil Oh has captured the best of both approaches. He's also snapped a healthy mix of personal style and brand devotion—as seen by the Rick Owens obsessives who wear him head-to-toe. Follow along as Phil Oh captures the best street style from the shows here.</p> <p>In Milan, thanks to Diesel's Glenn Martens, there's double the denim and double the fun. The streets are mixed with a bit of that old school Italian glamour mixed with the cool kids in their 1DR handbags and decked-out jackets. Other than denim, we're seeing lots of leather in all its incarnations—perhaps an influence of Bottega Veneta's Matthieu Blazy who's proven that anything can be leather. Follow along as Phil Oh captures the best street style from the city of Milan.</p> <p>How can you encapsulate the life and legacy of Vivienne Westwood—a chief genius in England's</p>	
640b207271d9faa9039ff97e	Phil Oh's Best Street Style Photos From the Fall 2023 Shows in Milan	<p>Photos From the Fall 2023 Shows in Milan</p>	

Lưu ý hình là Base64string, nên ta dùng cú pháp:

```

```

Như vậy khi tạo API lưu mới 1 Fashion thì ta chuyển hình qua Base64String

Bài 129 – Tao và triệu gọi API - HTTP GET – A Fashion

Yêu cầu:

-trong “my-server-mongodb” cập nhật index.js để cung cấp API lấy 1 Fashion khi biết Id

- trong “my-app” triệu gọi API để hiển thị chi tiết Fashion

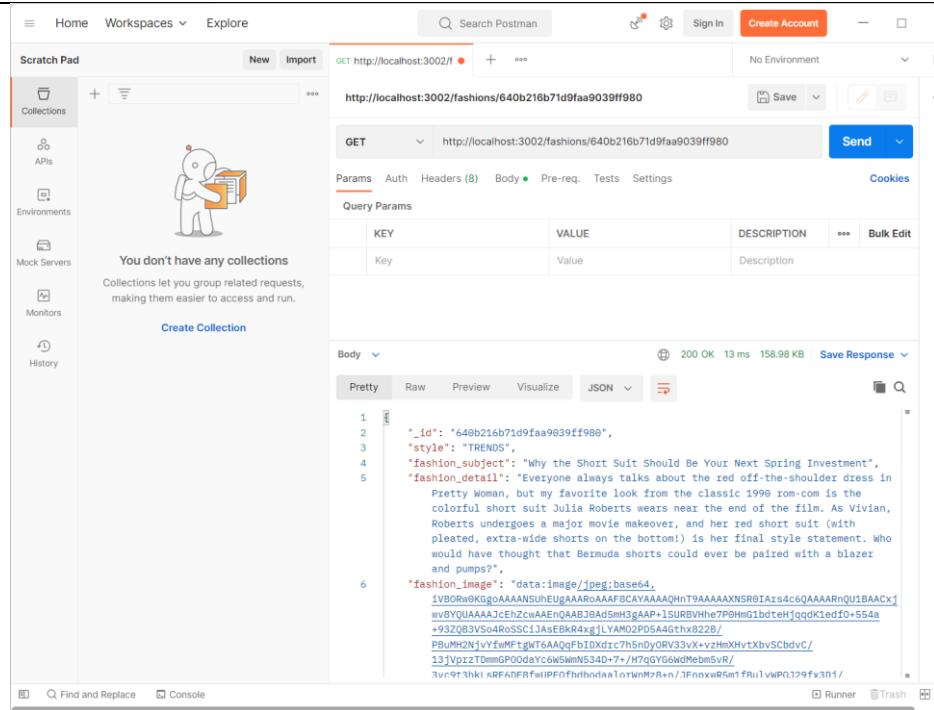
- Sinh viên suy nghĩ giao diện phù hợp để triệu gọi API

Hướng dẫn:

“index.js” trong “my-server-mongodb” bổ sung mã lệnh sau vào cuối tập tin:

```
app.get("/fashions/:id", cors(), async (req, res) => {
  var o_id = new ObjectId(req.params["id"]);
  const result = await fashionCollection.find({ _id: o_id }).toArray();
  res.send(result[0])
})
```

Dùng Postman để test:



Phần AngularJS Sinh viên xử lý tương tự như các hướng dẫn trước, giống như BookAPIService (hàm getBook(bookId:string)) và BookDetailComponent:

-Bước 1: Bổ sung hàm getFashion(id:string) trong lớp **FashionAPIService**

Hàm này gọi API Restful 1 Fashion

-Bước 2: Bổ sung component “FashionDetailComponent”, giao diện như bên dưới, tuy nhiên Sinh viên nên bố trí cho nó thẩm mỹ:

fashion-detail works!

Fashion ID:

Fashion Id: 640b226771d9faa9039ff983

Fashion Subject: Anne Hathaway Continues Her Bombshell Streak at Versace's L.A. Show

Celebrities were out in full force at Versace's fall 2023 show in Los Angeles on March 9, despite the last-minute decision to bring it forward by a day due to the weather forecast.

Detail: Naomi Campbell led a starry supermodel cast on the runway, while Lil Nas X, Miley Cyrus, Ariana DeBose, Elton John, and Dua Lipa lined the front row.



Bài 130 – Tao và triệu gọi API - HTTP POST – Create a Fashion**Yêu cầu:**

Xây dựng Restful API để thêm mới một Fashion, sử dụng AngularJS để triệu gọi API này, thiết kế giao diện phù hợp.

Hướng dẫn:

Bước 1: Trong “my-server-mongodb”, bổ sung API Post a Fashion trong file index.js (bổ sung cuối file):

```
app.post("/fashions", cors(), async(req,res)=>{
    //put json Fashion into database
    await fashionCollection.insertOne(req.body)
    //send message to client(send all database to client)
    res.send(req.body)
})
```

Lưu ý trong trường hợp dung lượng gửi lên Server (payload) quá lớn nó sẽ báo lỗi, ta cần xử lý lỗi này bằng cách bổ sung các lệnh sau ở trước các API, trong ví dụ này là cho payload 10mb, bạn có thể tăng lên nếu muốn:

```
const bodyParser=require("body-parser")
app.use(bodyParser.json({ limit: '10mb' }));
app.use(bodyParser.urlencoded({ extended: true, limit: '10mb' }));
app.use(express.json({ limit: '10mb' }));
app.use(express.urlencoded({ limit: '10mb' }));
app.use(express.json());
```

Mã lệnh đầy đủ của index.js:

```
const express = require('express');
const app = express();
const port = 3002;

const morgan=require("morgan")
app.use(morgan("combined"))

const bodyParser=require("body-parser")
app.use(bodyParser.json({ limit: '10mb' }));
app.use(bodyParser.urlencoded({ extended: true, limit: '10mb' }));

app.use(express.json({ limit: '10mb' }));
app.use(express.urlencoded({ limit: '10mb' }));
app.use(express.json());

const cors=require("cors");
app.use(cors())
```

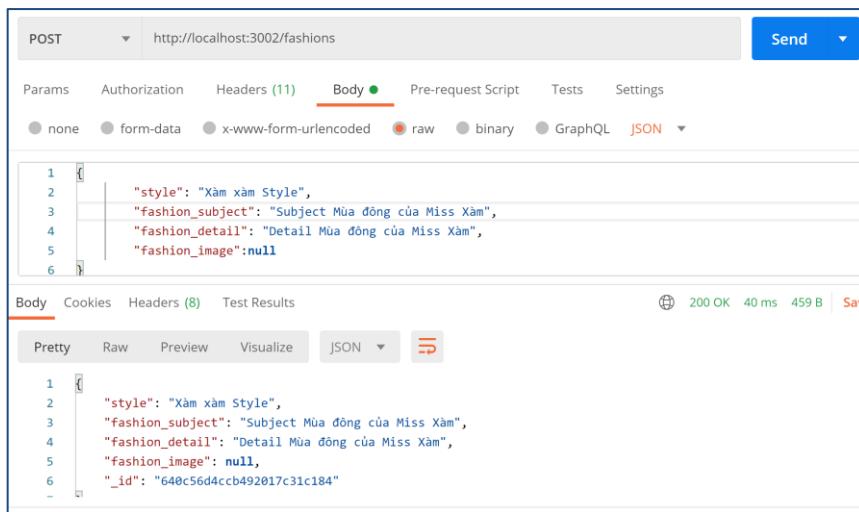
```
app.listen(port, ()=>{
    console.log(`My Server listening on port ${port}`)
})

app.get("/", (req, res)=>{
    res.send("This Web server is processed for MongoDB")
})

const { MongoClient, ObjectId } = require('mongodb');
client = new MongoClient("mongodb://127.0.0.1:27017");
client.connect();
database = client.db("FashionData");
fashionCollection = database.collection("Fashion");

app.get("/fashions", cors(), async (req, res)=>{
    const result = await fashionCollection.find({}).toArray();
    res.send(result)
})
app.get("/fashions/:id", cors(), async (req, res)=>{
    var o_id = new ObjectId(req.params["id"]);
    const result = await fashionCollection.find({_id:o_id}).toArray();
    res.send(result[0])
})
app.post("/fashions", cors(), async (req, res)=>{
    //put json Fashion into database
    await fashionCollection.insertOne(req.body)
    //send message to client(send all database to client)
    res.send(req.body)
})
```

Bước 2: Dùng postman để test, nếu nó thành công thì qua bước 3



Lưu ý đầu vào là 1 Json không có _id (vì nó tự động tạo ra), đầu ra chưong trình sẽ trả về cấu trúc input đồng thời có thêm _id cùng với giá trị của nó. Việc lập trình như vậy giúp chúng ta dễ dàng truy cập tới dữ liệu vừa đưa vào thông qua _id.

Input	Output
<pre>{ "style": "Xàm xàm Style", "fashion_subject": "Subject Mùa đông của Miss Xàm", "fashion_detail": "Detail Mùa đông của Miss Xàm", "fashion_image":null }</pre>	<pre>{ "style": "Xàm xàm Style", "fashion_subject": "Subject Mùa đông của Miss Xàm", "fashion_detail": "Detail Mùa đông của Miss Xàm", "fashion_image": null, "_id": "640c56d4ccb492017c31c184" }</pre>

Bước 3: Trong “my-app”, thêm hàm **postFashion** trong **FashionAPIService**

```
postFashion(aFashion:any):Observable<any>
{
  const headers=new HttpHeaders().set("Content-Type", "application/json; charset=utf-8")
  const requestOptions:Object={
    headers:headers,
    responseType:"text"
  }
  return
  this._http.post<any>("/fashions", JSON.stringify(aFashion),requestOptions).pipe(
    map(res=>JSON.parse(res) as Fashion),
    retry(3),
    catchError(this.handleError)
}
```

Bước 4: Bổ sung **FashionNewComponent** trong “my-app”, thiết kế giao diện và xử lý triệu gọi thêm mới một Fashion:

Fashion Style:

Fashion Subject:

Fashion Detail:

Fashion Image: No file chosen

File “fashion-new.component.html” thiết kế giao diện như sau:

```
<p>fashion-new works!</p>
<p>{{errMessage}}</p>
<table>
  <tr>
    <td>Fashion Style:</td>
```

```

<td>
    <input type="text" class="form-control" #name="ngModel" name="style"
id="style" placeholder="Nhập Style"
[(ngModel)]="fashion.style"/>

</td>
</tr>
<tr>
    <td>Fashion Subject:</td>
    <td>
        <input type="text" class="form-control" #name="ngModel"
name="fashion_subject" id="fashion_subject" placeholder="Nhập Subject"
[(ngModel)]="fashion.fashion_subject"/>
    </td>
</tr>
<tr>
    <td>Fashion Detail:</td>
    <td>
        <input type="text" class="form-control" #name="ngModel"
name="fashion_detail" id="fashion_detail" placeholder="Nhập Detail"
[(ngModel)]="fashion.fashion_detail"/>
    </td>
</tr>
<tr>
    <td>Fashion Image:</td>
    <td>
        <input type="file" class="file-input"
#name="ngModel" name="fashion_image" id="fashion_image"
[(ngModel)]="fashion.fashion_image"
(change)="onFileSelected($event,fashion)"
>
    </td>
</tr>
<tr>
    <td colspan="2">
        <button (click)="postFashion()">Submit</button>
    </td>
</tr>
</table>

```

File “fashion-new.component.ts” coding xử lý chuyển hình vật lý thành Base64String và đẩy lên Server như sau:

```

import { Component } from '@angular/core';
import { from } from 'rxjs';
import { FashionAPIService } from '../fashion-api.service';
import { Fashion } from '../models/Fashion';

@Component({
  selector: 'app-fashion-new',
  templateUrl: './fashion-new.component.html',
  styleUrls: ['./fashion-new.component.css']
})
export class FashionNewComponent {
  fashion=new Fashion();
  errMessage:string=''

```

```
constructor(private _service: FashionAPIService){  
  
}  
public setFashion(f:Fashion)  
{  
    this.fashion=f  
}  
onFileSelected(event:any,fashion:Fashion) {  
    let me = this;  
    let file = event.target.files[0];  
  
    let reader = new FileReader();  
    reader.readAsDataURL(file);  
    reader.onload = function () {  
        fashion.fashion_image=reader.result!.toString()  
    };  
    reader.onerror = function (error) {  
        console.log('Error: ', error);  
    };  
}  
  
postFashion()  
{  
    this._service.postFashion(this.fashion).subscribe({  
        next:(data)=>{this.fashion=data},  
        error:(err)=>{this.errorMessage=err}  
    })  
}  
}
```

Cáu hình routing hoặc tham chiếu trong **app.component.html** để chạy, ta có kết quả như mong muốn.

Bài 131 – Tạo và triệu gọi API - HTTP PUT – Update a Fashion

Yêu cầu:

Xây dựng API để cập nhật Fashion. Lưu ý bài này tương tự như bài tạo API thêm mới một Fashion.

Hướng dẫn:

- Moi thứ làm tương tự như bài thêm mới một Fashion.

Cập nhật “my-server-mongodb”, trong file index.js bổ sung API sau:

```
app.put("/fashions", cors(), async(req, res)=>{

    //update json Fashion into database
    await fashionCollection.updateOne(
        {_id:new ObjectId(req.body._id)}, //condition for update
        { $set: { //Field for updating
            style: req.body.style,
            fashion_subject:req.body.fashion_subject,
            fashion_detail:req.body.fashion_detail,
            fashion_image:req.body.fashion_image
        }
    }
)
//send Fahsion after updating
var o_id = new ObjectId(req.body._id);
const result = await fashionCollection.find({_id:o_id}).toArray();
res.send(result[0])
})
```

Hàm trên cập nhật thông tin của một Fashion, sau đó trả về Fashion đã được cập nhật. Ta dùng Postman để test nhằm đảm bảo mã lệnh được viết đúng:

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' selected, showing a message 'You don't have any collections'. The main area shows a 'PUT' request to 'http://localhost:3002/fashions/'. The 'Body' tab is selected, showing a JSON payload:

```

1
2
3
4
5
.... "style": "Xàm xàm 102 Style",
.... "fashion_subject": "Subject Mùa đông của Miss Xàm 102",
.... "fashion_detail": "Detail Mùa đông của Miss Xàm 102",
.... "fashion_image": "data:image/jpeg;base64,/9j/4AAQSKZJrgABAQEASABIAAD/
4aN2RXhpDZgAUTUAKgAAAAGAcWEPAIAAAAGAAAIngEQAATAAAACKAAIpaESAAMAAAABAAEAAAEEaA
AUAAAABAAAItgEbAAAABAAAItgEoAMAAAABAAIExAAIAAAA0AAA1vgITAAMAAAABAAEAAAExAAIAAAA0AAA
IvgEoAMAAAABAAIAAAExAAIAAAA0AAA1vgITAAMAAAABAAEAAAExAAIAAAA0AAA1vgITAAMAAAABAAEAAA
I

```

Below the body, the response is shown as a JSON object:

```

1
2
3
4
5
6
{
  "_id": "640d252af27a86da00cc06dd",
  "style": "Xàm xàm 102 Style",
  "fashion_subject": "Subject Mùa đông của Miss Xàm 102",
  "fashion_detail": "Detail Mùa đông của Miss Xàm 102",
  "fashion_image": "data:image/jpeg;base64,/9j/4AAQSKZJrgABAQEASABIAAD/
4aN2RXhpDZgAUTUAKgAAAAGAcWEPAIAAAAGAAAIngEQAATAAAACKAAIpaESAAMAAAABAAEAAAEEaA
AUAAAABAAAItgEbAAAABAAAItgEoAMAAAABAAIExAAIAAAA0AAA1vgITAAMAAAABAAEAAAExAAIAAAA0AAA
IvgEoAMAAAABAAIAAAExAAIAAAA0AAA1vgITAAMAAAABAAEAAAExAAIAAAA0AAA1vgITAAMAAAABAAEAAA
I

```

Các giao diện và service trong “my-app” làm tương tự như bài trước nên tài liệu không cung cấp chi tiết ở đây (sao chép qua rồi đổi chỗ gọi API thôi).

Bài 132 – Tao và triệu gọi API - HTTP DELETE – Remove a Fashion**Yêu cầu:**

Viết API xóa một Fashion khi biết _id

Hướng dẫn:

Trong “my-server-mongodb” bổ sung API DELETE ở cuối “index.js”

```
app.delete("/fashions/:id", cors(), async(req,res)=>{
    //find detail Fashion with id
    var o_id = new ObjectId(req.params["id"]);
    const result = await fashionCollection.find({_id:o_id}).toArray();
    //update json Fashion into database
    await fashionCollection.deleteOne(
        {_id:o_id}
    )
    //send Fahsion after remove
    res.send(result[0])
})
```

Test Postman để kiểm chứng mã lệnh có đúng không, nếu như màn hình bên dưới là đúng (chọn method DELETE):

The screenshot shows the Postman interface with the following details:

- Request URL:** http://localhost:3002/fashions/640d251bf27a86da80cc06dc
- Method:** DELETE
- Response Body (Pretty JSON):**

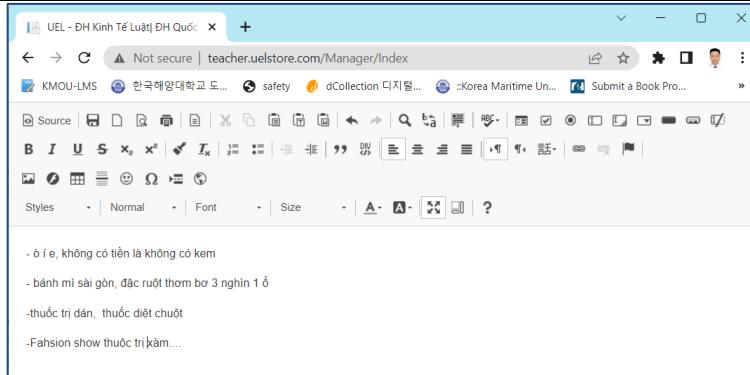
```
_id: "640d251bf27a86da80cc06dc",
style: "Xàm xàm Style",
fashion_subject: "Subject Mùa đông của Miss Xàm",
fashion_detail: "Detail Mùa đông của Miss Xàm",
fashion_image: null
```
- Status:** 200 OK
- Time:** 15 ms
- Size:** 459 B

Phần “my-app” là tương tự như BookDeleteComponent và BookAPIService (deleteBook(bookId:string):Observable<any>). Tài liệu không trình bày chi tiết phần triệu gọi API xóa này.

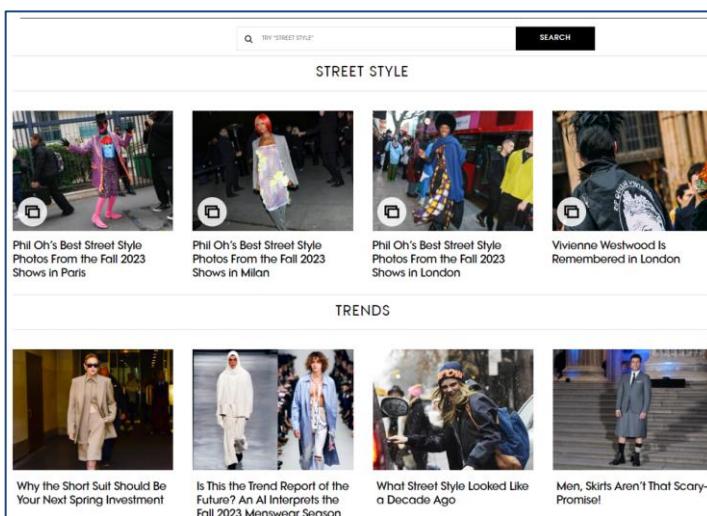
Bài 133 – Bài tập tổng hợp(*)**Yêu cầu:**

Áp dụng các bài học trong Module này để Xây dựng Website Fashion tổng hợp theo mô tả:

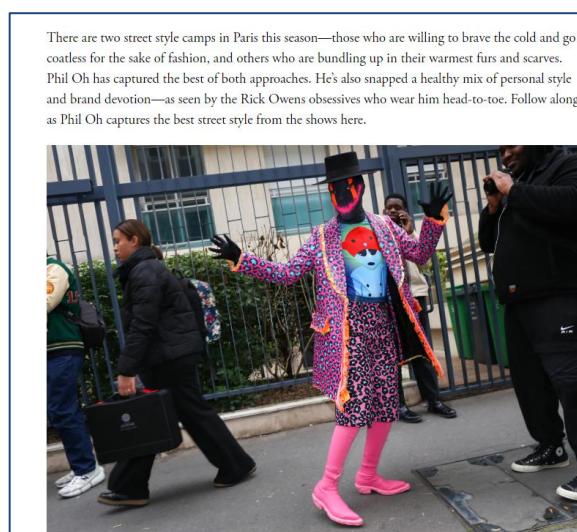
- Cơ sở dữ liệu MongoDB tên **FashionData**, Sinh viên cần suy luận và tự đề xuất ra cấu trúc dữ liệu phù hợp để thiết kế trong MongoDB:
 - o Có một collection duy nhất tên là **Fashion** gồm các thông tin: ObjectId (MongoDB tự động tạo ra), tiêu đề fashion, chi tiết fashion, hình thumbnail, Style của Fashion, ngày tạo.
 - o Nhập dữ liệu mẫu khoảng 3 Style, mỗi Style có khoảng 3 tới 5 fashion.
- Module Restful API (NodeJS, ExpressJS), bao gồm các API (tạo một project tên “**server-fashion**”, port 4000):
 - o API trả về toàn bộ Fashion, sắp xếp theo ngày tạo giảm dần
 - o API lọc danh sách các Fashion theo 1 Style nào đó
 - o API trả về một Fashion dựa vào ObjectId (gọi chung là id)
 - o API thêm mới một Fashion
 - o API chỉnh sửa một Fashion
 - o API xóa một Fashion dựa vào id
- Module Admin (AngularJs Front End) để quản trị hệ thống, bao gồm các chức năng (tạo một project tên “**admin-fashion**” , port 4001):
 - o Xem toàn bộ danh sách Fashion, mỗi dòng có nút xem chi tiết, nút sửa, và nút xóa (phải xác nhận trước khi xóa).
 - o Ngoài ra trong trang xem danh sách Fashion có nút Thêm mới một Fashion.
 - o Khi nhấn nút sửa hay nút thêm mới đều hiển thị một màn hình chi tiết để nhập thông tin cho Fashion, sau khi lưu thành công thì quay trở lại màn hình xem danh sách Fashion, yêu cầu phải sử dụng **WYSIWYG** (What You See Is What You Get) Html Editor cho field **chi tiết fashion**, Sinh viên tìm các công cụ tương tự:



- Module Client (AngularJs Front End) để trình diễn, bao gồm các chức năng(tạo một project tên “**client-fashion**” , port 4002):
 - o Hiển thị Fashion theo nhóm Style như hình dưới đây, hình đại diện là thuộc tính thumbnail.



- o Khi nhấp vào từng Fashion thì hiển thị chi tiết, hình và nội dung nếu có nó thuộc field chi tiết được nhập bằng **WYSIWYG**:



- o Tìm kiếm Fashion theo Style, người dùng nhập Style hoặc chọn từ dropdownlist thì lọc ra các Fashion thuộc Style đã chọn.

Module 13 – Cookie và Session

Nội dung kiến thức thực hành:

- + Hiểu được ý nghĩa của Cookie và Session
- + Lập trình Cookie để lưu trữ dữ liệu phía client
- + Lập trình Session để lưu trữ dữ liệu phía Server
- + Tạo giỏ hàng với Session

Bài 134 – Trình bày ý nghĩa của Cookie và Session

Yêu cầu:

Hãy trình bày ý nghĩa của Cookie và Session trong các hệ thống Website thương mại điện tử. So sánh sự khác biệt về cơ chế hoạt động của 2 đối tượng này.

Liệt kê một số thư viện hỗ trợ xử lý Cookie và Session.

Hướng dẫn:

- Sử dụng thư viện **cookie-parser** để minh xử lý Cookie
- Sử dụng thư viện **express-session** để xử lý Session

Bài 135 – Lập trình Cookie

Yêu cầu:

Sử dụng cookie-parser minh họa các chức năng:

- Khởi tạo đối tượng Cookie
- Thêm mới Cookie
- Truy suất Cookie
- Xóa Cookie

Hướng dẫn:

Bước 1:

- Mở project “**my-server-mongodb**”
- Sau đó cài đặt **cookie-parser** bằng lệnh:

npm install --save cookie-parser

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

● PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server-mongodb> npm install --save cookie-parser
added 2 packages, removed 1 package, and audited 120 packages in 865ms

10 packages are looking for funding
  run `npm fund` for details

  found 0 vulnerabilities
○ PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server-mongodb>
```

Bước 2:

Hiệu chỉnh “index.js” để cài đặt khởi tạo đối tượng Cookie:

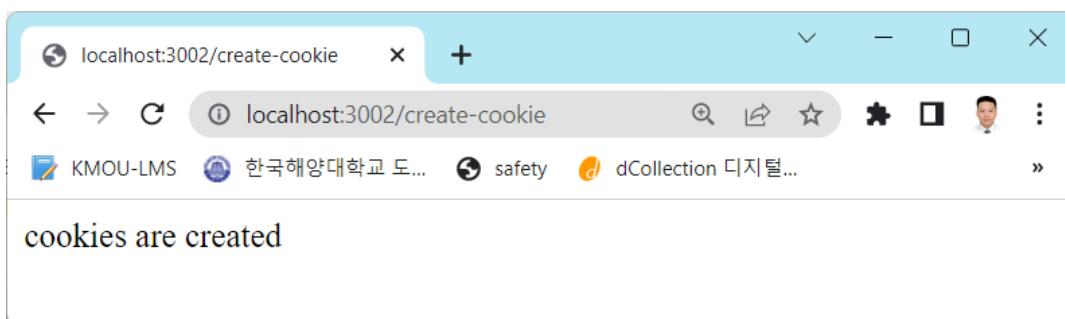
```
var cookieParser = require('cookie-parser');
app.use(cookieParser());
```

Bước 3:

Mình họa 1 API để tạo Cookie, dữ liệu đơn và dữ liệu JsonObject:

```
app.get("/create-cookie", cors(), (req, res) =>{
  res.cookie("username", "tranduythanh")
  res.cookie("password", "123456")
  account={"username": "tranduythanh",
            "password": "123456"}
  res.cookie("account", account)
  res.send("cookies are created")
})
```

Khởi tạo server và kiểm thử api create-cookie, ta có kết quả:

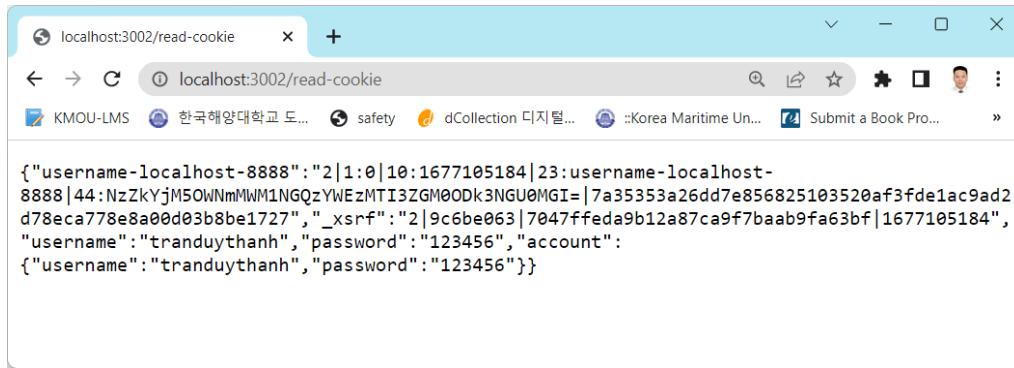


Bước 4:

Mình họa 1 API để đọc Cookie:

```
app.get("/read-cookie", cors(), (req, res) =>{
  //cookie is stored in client, so we use req
  res.send(req.cookies)
})
```

Chạy chương trình và gọi api read-cookie ta có kết quả:



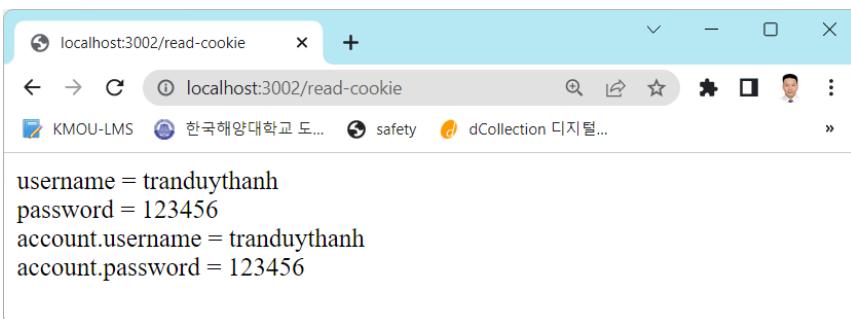
Quan sát kết quả, rõ ràng nó là một JsonObject, ta có thể dễ dàng truy cập tới từng Cookie mà ta đã lưu trước đó.

Bước 5:

Hiệu chỉnh api read-cookie để truy suất và bóc tách ra từng biến cookie mà ta đã lưu (chú ý dùng req):

```
app.get("/read-cookie", cors(), (req, res) => {
    //cookie is stored in client, so we use req
    username = req.cookies.username
    password = req.cookies.password
    account = req.cookies.account
    infor = "username = " + username + "<br/>"
    infor += "password = " + password + "<br/>"
    infor += "account.username = " + account.username + "<br/>"
    infor += "account.password = " + account.password + "<br/>"
    res.send(infor)
})
```

Chạy chương trình và gọi api read-cookie ta có kết quả:



Ngoài ra để tạo cookie với timeout (cookie có thời hạn) thì ta có thể code:

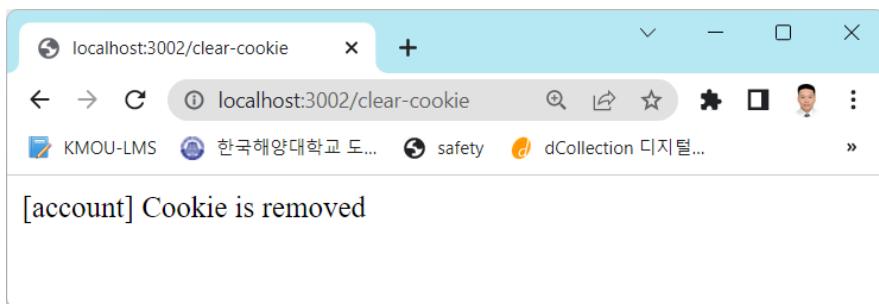
```
//Expires after 360000 ms from the time it is set.
res.cookie("infor_limit1", 'I am limited Cookie - way 1', {expire: 360000 +
Date.now()});
res.cookie("infor_limit2", 'I am limited Cookie - way 2', {maxAge: 360000});
```

Bước 6:

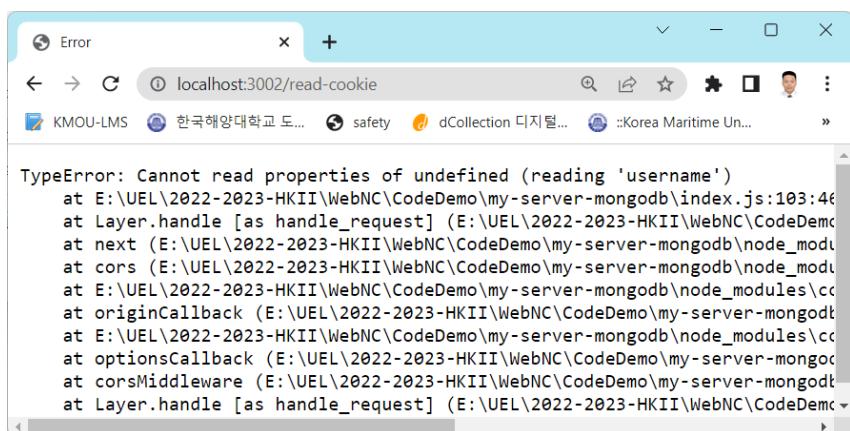
Thêm api clear-cookie để xóa Cookie đã lưu (chú ý dùng **res**):

```
app.get("/clear-cookie", cors(), (req, res) =>{
    res.clearCookie("account")
    res.send("[account] Cookie is removed")
})
```

Chạy api ta có kết quả:



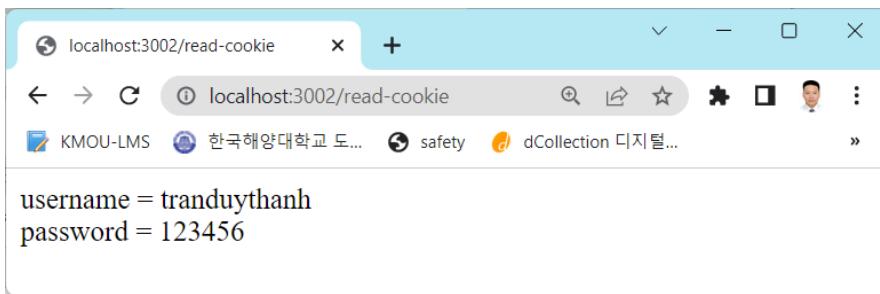
Lưu ý sau khi xóa Cookie mà ta có tính truy suất thì nó bị null, và báo lỗi:



Do đó ta cần phải kiểm tra Cookie đọc ra, nếu nó null thì không xử lý, hiệu chỉnh lại:

```
app.get("/read-cookie", cors(), (req, res) =>{
    //cookie is stored in client, so we use req
    username = req.cookies.username
    password = req.cookies.password
    account = req.cookies.account
    infor = "username = " + username + "<br/>"
    infor += "password = " + password + "<br/>"
    if(account != null)
    {
        infor += "account.username = " + account.username + "<br/>"
        infor += "account.password = " + account.password + "<br/>"
    }
    res.send(infor)
})
```

Kết quả sau khi hiệu chỉnh kiểm tra null:



Lúc này không còn bị báo lỗi nữa, chương trình chỉ hiển thị cookie của username và password, còn cookie account thì bị xóa rồi, nó bị null nên không hiển thị.

Lưu ý Cookie là lưu phía client (lưu phía trình duyệt), chạy trình duyệt nào thì cookie lưu theo trình duyệt đó và không hiểu các trình duyệt khác. Ví dụ nếu chạy Chrome và lưu cookie bên Chrome thì bên Firefox hay Microsoft Edge không có truy suất được. Và nhớ rằng cho dù lập trình Cookie có đúng nhưng nếu người sử dụng cầm mọi trình duyệt lưu cookie thì các mã lệnh của ta cũng vô nghĩa.

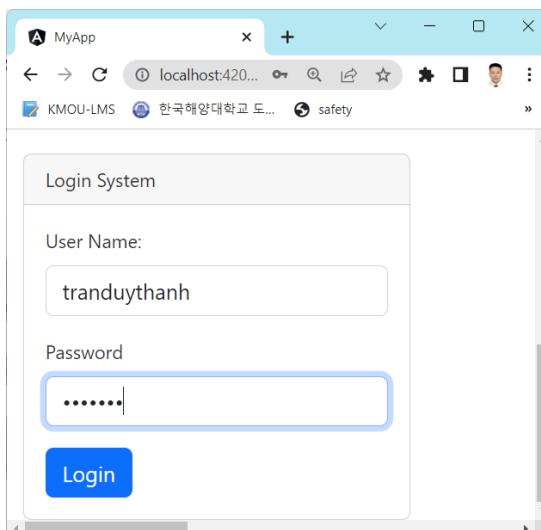
Bài 136 – Lập trình Cookie – Lưu thông tin đăng nhập

Yêu cầu:

Trong cơ sở dữ liệu “FashionData”, bổ sung thêm 1 collection tên là “User”, nó có 2 field là “username” và “password”. Hãy viết các Restful API để đăng nhập, nếu đăng nhập thành công thì lưu Cookie thông tin đăng nhập, các lần sau mở trình duyệt lên thì đọc cookie và hiển thị các thông tin này lên các ô nhập user name và password.

Hướng dẫn:

Tạo LoginComponent, thiết kế giao diện như hình dưới đây:



Dưới đây là mã lệnh của “login.component.html”:

```
<div class="container mt-4">
  <div class="card" style="width: 18rem;">
    <div class="card-header">Login System</div>
    <div class="card-body">
      <form action="/auth/register" method="POST">
        <div class="mb-3">
          <label for="name-reg" class="form-label">User Name:</label>
          <input type="text" class="form-control" id="name-reg" name="name">
        </div>
        <div class="mb-3">
          <label for="password-reg" class="form-label">Password</label>
          <input type="password" class="form-control" id="password-reg" name="password">
        </div>

        <button type="submit" class="btn btn-primary">Login</button>
      </form>
    </div>
  </div>
</div>
```

Yêu cầu Sinh viên áp dụng các bài học trước để tạo Collection “User”, nhập một vài Document mẫu, viết API Restful đăng nhập trong file **index.js** của dự án “**my-server-mongodb**”. API đăng nhập nên dùng phương thức POST, Khi đăng nhập thành công thì lưu Cookie, lần mở màn hình Login tiếp theo sẽ đọc cookie và hiển thị lên giao diện.

Bài 137 – Lập trình Session

Yêu cầu:

Sử dụng **express-session** để minh họa cách sử dụng Session cơ bản.

Hướng dẫn:

Bước 1:

- Mở project “**my-server-mongodb**”
- Sau đó cài đặt **express-session** bằng lệnh:

```
npm install --save express-session
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

● PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server-mongodb> npm install --save express-session
added 4 packages, and audited 124 packages in 3s

10 packages are looking for funding
  run `npm fund` for details

  found 0 vulnerabilities
○ PS E:\UEL\2022-2023-HKII\WebNC\CodeDemo\my-server-mongodb>
```

Bước 2:

Hiệu chỉnh “index.js” để cài đặt khởi tạo đối tượng Session:

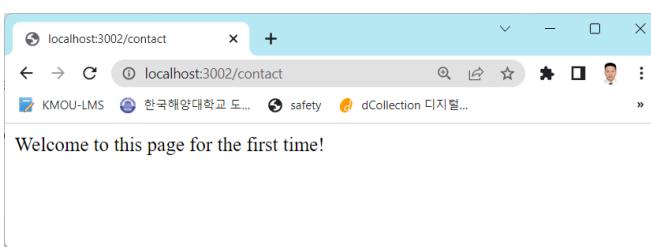
```
var session = require('express-session');
app.use(session({secret: "Shh, its a secret!"}));
```

Bước 3:

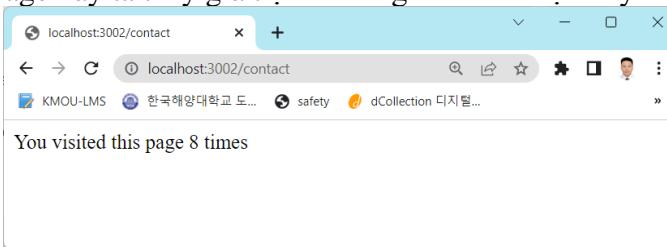
Mình họa 1 API để tạo Session visited

```
app.get("/contact", cors(), (req, res) => {
  if(req.session.visited!=null)
  {
    req.session.visited++
    res.send("You visited this page "+req.session.visited +" times")
  }
  else
  {
    req.session.visited=1
    res.send("Welcome to this page for the first time!")
  }
})
```

Khởi tạo server và kiểm thử api **contact**, ta có kết quả:



Nhấn F5 nhiều lần ở page này ta thấy giá trị lưu trong session được thay đổi:



Lưu ý tương tự Cookie, ta có thể lưu cấu trúc dữ liệu bất kỳ cho Session.

Bài 138 – Lập trình Session – Bài tập tổng hợp Shopping Cart(*)**Yêu cầu:**

Sử dụng Session minh họa xử lý giỏ hàng. Trong các trang thương mại điện tử, khi người sử dụng chọn các mặt hàng và đưa vào giỏ hàng (chưa thực sự mua) thì ta dùng Session để xử lý lưu trữ tạm thời. Hoặc chúng ta vẫn lưu vào cơ sở dữ liệu ở bảng tạm để xử lý trường hợp khách hàng đặt hàng dở dang: Đang đặt thì tạm dừng, hoặc đặt xong tới khâu thanh toán thì tạm hoãn... mọi thao tác ta cần lưu trữ lại để chăm sóc khách hàng (ví dụ khách hàng X đặt 5 món hàng nhưng tình trạng đơn hàng vẫn chưa hoàn tất, sau một thời gian kiểm tra hệ thống thì chủ cửa hàng nên chủ động liên hệ để push đơn hàng).

Trong bài này, yêu cầu Sinh viên chỉ sử dụng Session để lưu dữ liệu tạm thời các thao tác của khách hàng trên Server (bộ nhớ Server). Chỉ khi nào khách hàng xác nhận thanh toán thì mới lưu vào cơ sở dữ liệu các lựa chọn có trong đơn hàng. Trường hợp này thì khi tắt hệ thống Website thì các dữ liệu của khách hàng đang lưu trữ trong session thường sẽ bị mất hoặc sau thời gian timeout nó cũng bị mất. Yêu cầu Sinh viên thực hiện và minh họa các mô tả sau:

- Trong MongoDB tạo một collection Product, các Field tùy thuộc vào nhu cầu thiết kế của sinh viên để tạo cho phù hợp theo ý mình
- Viết các API để xử lý hiển thị Product, cho các khách lựa chọn Product và lưu vào Session (Session này tạo 1 biến đối tượng là **cart** để lưu JSONArray các Product mà khách hàng lựa chọn)
- Thiết kế giao diện cho khách hàng lựa chọn sản phẩm để đưa vào **cart**.
- Thiết kế giao diện để hiển thị chi tiết Product trong **cart**

Hướng dẫn:

Ví dụ màn hình hiển thị Product để khách hàng chọn mua:



Diamond Promise Ring 1/6 ct tw

Round-cut 10K White Gold

\$399.99

[ADD TO CART](#)

Diamond Promise Ring 1/4 ct tw

Round/Baguette 10K White Gold

\$529.00

[ADD TO CART](#)

Diamond Promise Ring 1/6 ct tw

Black/White Sterling Silver

\$159.00

[ADD TO CART](#)



Diamond Promise Ring 1/5 ct tw

Round-cut Sterling Silver

\$289.00

[ADD TO CART](#)

Diamond Promise Ring 1/5 ct tw

Round-cut Sterling Silver

\$289.00

[ADD TO CART](#)

Diamond Promise Ring 1/8 ct tw

Round-cut Sterling Silver Ring

\$229.00

[ADD TO CART](#)

Mỗi lần bấm “**ADD TO CART**” thì đưa Product vào giỏ hàng (biến cart)

Khi vào chức năng xem giỏ hàng, hiển thị giao diện như bên dưới:

Shopping cart

Remove	SKU	Image	Product(s)	Price	Qty.	Total
<input type="checkbox"/>			Diamond Promise Ring 1/5 ct tw Round-cut Sterling Silver	\$289.00	<input type="text" value="1"/>	\$289.00
<input type="checkbox"/>			Diamond Promise Ring 1/5 ct tw Round-cut Sterling Silver	\$289.00	<input type="text" value="1"/>	\$289.00
<input type="checkbox"/>			Diamond Promise Ring 1/8 ct tw Round-cut Sterling Silver Ring	\$229.00	<input type="text" value="1"/>	\$229.00

[Update shopping cart](#)

[Continue shopping](#)

-Cho phép checked Product để xóa khỏi giỏ hàng, cập nhật số lượng → bấm “Update shopping cart” để cập nhật giỏ hàng.

-Muốn mua tiếp (quay trở lại màn hình Product) → bấm “Continue shopping”