



โครงการ Sprout FTP เว็บฝากไฟล์รูปภาพ

เสนอ

ผู้ช่วยศาสตราจารย์ดร.สวิน วงศ์ประเมษฐ์

จัดทำโดย

นายวีรพงศ์ วสุมงคลพจน์ 670112418022 นักศึกษา ไอที ปี 2 หมู่ที่ 1

โครงการนี้เป็นส่วนหนึ่งของรายวิชา เทคโนโลยีอินเทอร์เน็ต (4132203)

ภาคเรียนที่ 2/2568

สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์ มหาวิทยาลัยราชภัฏบุรีรัมย์

คำนำ

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา เทคโนโลยีอินเทอร์เน็ต (4132203) จัดทำขึ้นเพื่อศึกษาและพัฒนาระบบ FTP (File Transfer Protocol) สำหรับการส่งไฟล์ โดยมุ่งเน้นให้ผู้ใช้สามารถฝากไฟล์ได้สะดวกยิ่งขึ้น

เนื้อหาประกอบด้วย ที่มาและความสำคัญ หลักการทำงาน เครื่องมือที่ใช้ และผลการดำเนินงาน โดยนำเทคโนโลยีสำคัญ เช่น HTML , CSS , C# (Blazor) , ไลบรารี FluentFTP และ Docker มาสนับสนุนการอัปโหลด ดาวน์โหลด และลบข้อมูลผ่านเว็บไซต์

คณะผู้จัดทำหวังว่ารายงานนี้จะเป็นประโยชน์แก่ผู้ที่สนใจศึกษาเทคโนโลยีอินเทอร์เน็ต และสามารถนำความรู้ไปประยุกต์ใช้ในการพัฒนาระบบอื่น ๆ ต่อไป

ลงชื่อคณะผู้จัดทำ

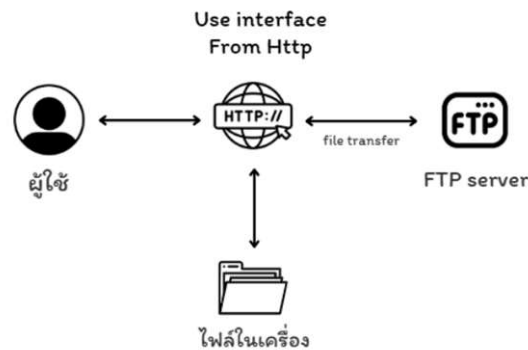
นายวีรพงศ์ วสุมงคลพจน์

วันที่ 11 พฤศจิกายน 2568

เนื้อหา

บริการที่ได้รับคืออะไร : บริการ FTP (File Transfer Protocol)

FTP คืออะไร? : File Transfer Protocol หรือ FTP คือโพรโทคอลมาตรฐานบนเครือข่ายคอมพิวเตอร์ที่ถูกออกแบบมาเพื่อใช้ในการแลกเปลี่ยนและจัดการไฟล์ระหว่างเครื่องคอมพิวเตอร์แม่ข่าย (Server) และเครื่องคอมพิวเตอร์ลูกข่าย (Client) โดยทำงานในลักษณะ Client-Server Model กล่าวคือ Client จะเป็นฝ่ายส่งคำร้องขอ (Request) ไปยัง Server เช่น ขอรายชื่อไฟล์, ขออัปโหลดไฟล์ หรือขอดาวน์โหลดไฟล์ และ Server จะเป็นฝ่ายตอบสนอง (Response) ต่อคำร้องขอนั้นๆ



ประวัติความเป็นมา : FTP ถือเป็นหนึ่งในโพรโทคอลรุ่นแรกๆ ของอินเทอร์เน็ต มีการเผยแพร่ข้อกำหนดทางเทคนิค (RFC 114) ครั้งแรกในปี ค.ศ. 1971 ทำให้ FTP มีอายุมากกว่าโพรโทคอล HTTP ที่ใช้สำหรับเปิดเว็บไซต์ในปัจจุบันเสียอีก ด้วยความเรียบง่ายและประสิทธิภาพในการถ่ายโอนไฟล์ขนาดใหญ่ ทำให้ FTP ยังคงได้รับความนิยมใช้งานในหลายสถานการณ์ เช่น การอัปโหลดไฟล์เว็บไซต์ขึ้นสู่ Web Hosting, การสำรองข้อมูล หรือการเป็นพื้นที่กลางสำหรับแลกเปลี่ยนไฟล์ในองค์กร

ขอบเขตของโครงการ

1. ผู้ใช้ระบบ

1.1 ผู้ใช้ทั่วไป

- 1.1.1 สามารถอัปโหลดไฟล์จากเครื่องคอมพิวเตอร์ขึ้นไปยังเซิร์ฟเวอร์ FTP ผ่านหน้าเว็บ
- 1.1.2 สามารถดาวน์โหลดไฟล์จากเซิร์ฟเวอร์ FTP มายังเครื่องคอมพิวเตอร์ของตนเอง
- 1.1.3 สามารถดูรายการไฟล์และโฟลเดอร์ทั้งหมดที่เก็บอยู่ในเซิร์ฟเวอร์ FTP
- 1.1.4 สามารถจัดการไฟล์ได้ เช่น ลบไฟล์ เปลี่ยนชื่อไฟล์ หรือสร้างโฟลเดอร์ใหม่

2. ฟังก์ชันหลัก

2.1 อัปโหลดไฟล์

- 2.1.1 ให้ผู้ใช้เลือกไฟล์จากเครื่องและอัปโหลดขึ้นไปเก็บบนเซิร์ฟเวอร์

2.2 ดาวน์โหลดไฟล์

- 2.2.1 ให้ผู้ใช้สามารถเลือกไฟล์ที่ต้องการและดาวน์โหลดลงมายังเครื่องตนเอง

2.3 ดูรายการไฟล์

- 2.3.1 แสดงรายการไฟล์และโฟลเดอร์ในรูปแบบตารางหรือลิสต์

2.4 การจัดการไฟล์

- 2.4.1 ผู้ใช้สามารถลบหรือเปลี่ยนชื่อไฟล์/โฟลเดอร์

เครื่องมือที่ใช้งาน :

1. **C#** เป็นภาษาโปรแกรมเชิงวัตถุที่ทันสมัย พัฒนาโดย Microsoft และทำงานบน .NET Framework โดยใช้สร้างแอปพลิเคชันได้หลากหลายประเภท เช่น แอปพลิเคชันบนเดสก์ท็อป, เว็บ, โมบายล์, เกม และบริการบนคลาวด์ C# มีพื้นฐานมาจากตระกูลภาษา C และมีความใกล้เคียงกับภาษาอื่น ๆ อย่าง C++ และ Java ทำให้ผู้ที่มีความรู้เกี่ยวกับภาษานั้น ๆ สามารถเรียนรู้ได้ง่าย



2. **HTML** (HyperText Markup Language) เป็นภาษามาร์กอัปที่ใช้ในการสร้างและจัดโครงสร้าง

เนื้อหาบนหน้าเว็บ HTML ทำหน้าที่กำหนดองค์ประกอบต่างๆ บนหน้าเว็บ เช่น ข้อความ รูปภาพ ลิงก์ ปุ่ม และฟอร์ม โดยทำงานร่วมกับ CSS (Cascading Style Sheets) สำหรับการออกแบบ



3. **CSS** (Cascading Style Sheets) เป็นภาษาที่ใช้สำหรับกำหนดรูปแบบและตกแต่งหน้าเว็บ โดยทำงานร่วมกับ HTML เพื่อให้หน้าเว็บดูสวยงามและใช้งานง่าย CSS ช่วยให้สามารถควบคุม สี ขนาด ตัวอักษร ระยะห่าง การจัดวาง และเอฟเฟกต์ต่างๆ ได้โดยไม่ต้องแก้ไขโค้ด HTML



4. **FluentFTP** ไลบรารีไคลเอนต์ FTP/FTPS ที่เขียนด้วยภาษา C# สำหรับ .NET ซึ่งมีคุณสมบัติครบถ้วนและใช้งานง่าย ทำให้สามารถเขียนโปรแกรมเพื่อเชื่อมต่อกับเซิร์ฟเวอร์ FTP และ FTPS เพื่อจัดการไฟล์ได้สะดวก สามารถใช้ในแอปพลิเคชันได้ทั้งแบบส่วนตัวและแบบโอเพนซอร์ส



nuget v33.0.3 downloads 4.2M contributors 53 code quality A

FluentFTP is a fully managed FTP and FTPS library for .NET
FTP commands, File uploads/downloads, SSL/TLS connecti

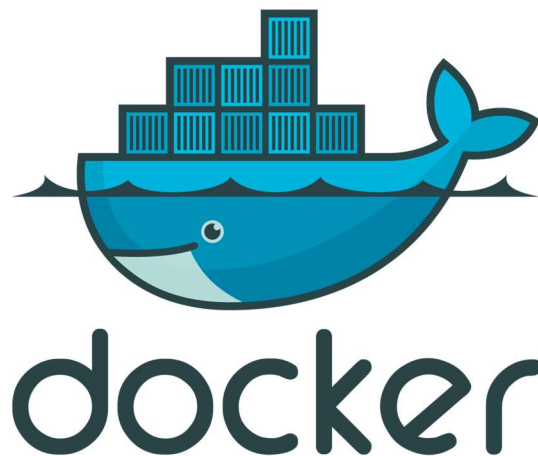
5. ASP.NET Razor Pages คือ

เฟรมเวิร์กการพัฒนาเว็บใน ASP.NET Core ที่ช่วยให้ นักพัฒนาสร้างหน้าเว็บแบบไดนามิกโดยรวมโค้ดฝั่งเซิร์ฟเวอร์ (C#) เข้ากับ HTML ได้โดยตรง ผ่านไวยากรณ์ที่เรียกว่า Razor ซึ่งมีลักษณะคล้ายกับการรวมไฟล์ .aspx ของ ASP.NET Web Forms และไฟล์ .cshtml ซึ่งมีส่วนของ C# และ HTML อยู่ในไฟล์เดียวกัน



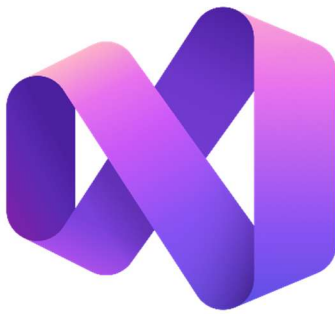
6. Docker เป็นแพลตฟอร์มสำหรับ จำลองและจัดการสภาพแวดล้อมการทำงานของแอปพลิเคชัน

โดยใช้ Container ซึ่งช่วยให้สามารถพัฒนา รัน และจัดการแอปพลิเคชันได้อย่างสะดวกและรวดเร็ว ไม่ต้องกังวลเรื่องความแตกต่างของระบบปฏิบัติการหรือการตั้งค่าระบบ



7. Visual Studio

Microsoft Visual Studio ซึ่งเป็น Integrated Development Environment (IDE) หรือชุดพัฒนาโปรแกรมที่ใช้สำหรับสร้างและแก้ไขซอฟต์แวร์ เป็นโปรแกรมที่ช่วยให้นักพัฒนาเขียนโค้ด แก้ไข ดีบั๊ก และเผยแพร่แอปพลิเคชันได้อย่างรวดเร็วและมีประสิทธิภาพ



การทำงานและพอร์ตที่ใช้งาน

Port : เราได้กำหนดให้ Server สามารถ 0-21110" ใน Docker Compose ก็คือการเปิดทั้ง "ห้องสั่งการ" (พอร์ต 2 เลือกใช้พอร์ตในช่วง 21100 ถึง 21110 สำหรับการส่งข้อมูลนี้) และเตรียม "ประตูขนส่งสินค้า" (พอร์ต 21100-21110)

1. พอร์ต 21: พอร์ตควบคุม (Control Port)

- เปรียบเสมือน "สมอง" หรือ "ผู้จัดการ" ของการเชื่อมต่อ
- ทำหน้าที่รับ-ส่ง คำสั่ง เท่านั้น ไม่ใช่ตัวไฟล์
- ตัวอย่างคำสั่ง:
 - USER/PASS: การยืนยันตัวตน (Login)
 - LIST: ขอรายชื่อไฟล์และโฟลเดอร์
 - STOR: คำสั่ง "เตรียมอัปโหลดไฟล์"
 - RETR: คำสั่ง "เตรียมเป็นที่แรกเสมอ เพื่อส่ง "คำสั่ง" เท่านั้น เช่น:
 - USER / PASS : สำหรับ Login
 - LIST : ขอดูรายชื่อไฟล์
 - STOR : บอกว่า "ฉันกำลังจะอัปโหลดไฟล์นะ"
 - RETR : บอกว่า "ฉันต้องการดาวน์โหลดไฟล์นี้"
- ข้อมูลที่วิ่งในช่องทางนี้จะมีขนาดเล็กมาก

2. พอร์ต Data: ช่องทางสำหรับข้อมูล (Data Port)

- **หน้าที่:** เปรียบเสมือน "ประตูขนส่งสินค้า" หรือ "ช่องทางด่วนพิเศษ"
- **สิ่งที่ทำ:** ใช้สำหรับรับ-ส่ง "ข้อมูลของไฟล์จริงๆ" ทั้งหมด ช่องทางนี้จะถูกเปิดขึ้นมาใช้ชั่วคราวเมื่อมีการอัปโหลดหรือดาวน์โหลดเกิดขึ้นเท่านั้น

แล้วสองช่องทางนี้ทำงานร่วมกันอย่างไร?

โหมดที่เรียกว่า **Passive Mode** ซึ่งเป็นโหมดที่นิยมและปลอดภัยที่สุด มีขั้นตอนง่ายๆ ดังนี้:

1. โปรแกรมของเรา (Client) ติดต่อไปที่ **พอร์ต 21** ของ Server เพื่อ Login และส่งคำสั่ง "ฉันต้องการอัปโหลดไฟล์"
2. Server ตอบกลับมาทางพอร์ต 21 ว่า "ได้เลย, ให้ไปที่ **ประตูขนส่งหมายดาวโหลดไฟล์"

พอร์ตข้อมูล (Data Port)

- เปรียบเสมือน "ท่อส่งของขนาดใหญ่" หรือ "รถบรรทุก"
 - ทำหน้าที่ รับ-ส่งข้อมูลของไฟล์จริงๆ
 - เป็นพอร์ตที่ **Server** กำหนดขึ้นมาใหม่ ทุกครั้งที่มีการส่งไฟล์ (Dynamic Port) ผ่านเปิดการเชื่อมต่อใหม่ไปยัง **พอร์ต 21105** ของ Server แล้วเริ่มส่งข้อมูลไฟล์ทั้งหมดผ่านช่องทางนี้
1. เมื่อส่งไฟล์เสร็จ ประตูขนส่งหมายเลข 21105 ก็จะถูกปิดลง
- ในโปรเจกต์ของ เราได้กำหนดให้ Server สามารถ 0-21110" ใน Docker Compose ก็คือการเปิดทั้ง "ห้องส่งการ" (พอร์ต 2 เลือกใช้พอร์ตในช่วง 21100 ถึง 21110 สำหรับการส่งข้อมูลนี้1) และเตรียม "ประตูขนส่งสินค้า" (พอร์ต 21100-21110)

FTP Active Mode

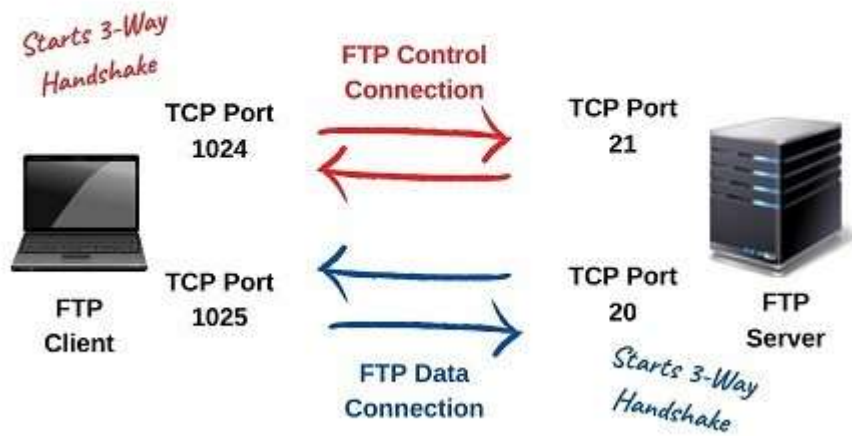
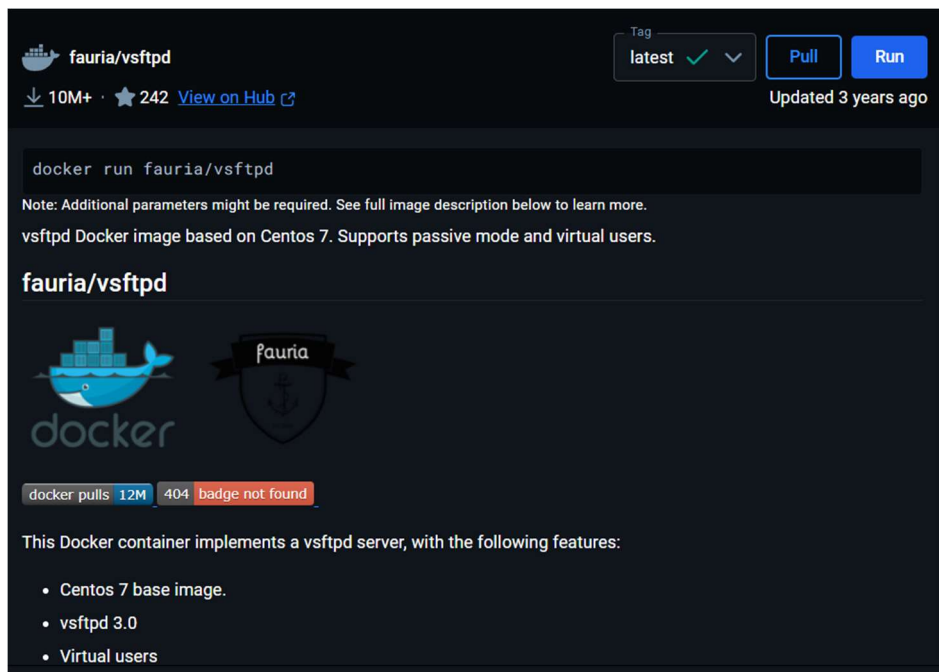


Image บน docker hub

เราได้เลือกใช้ Image ชื่อ **fauria/vsftpd** ซึ่งเป็น Image ที่บรรจุ FTP Server ที่ชื่อว่า vsftpd (Very Secure FTP Daemon) เอาไว้เรียบร้อยแล้ว การใช้ Image สำเร็จรูปเช่นนี้ช่วยลดเวลาในการติดตั้งและตั้งค่า Server ลงไปได้อย่างมหาศาล เราสามารถดึง (pull) Image และรันเป็น Container ได้ด้วยคำสั่งเพียงไม่กี่บรรทัด



ตัวอย่าง Docker-compose

```
version: '3.8'

services:
  # Service ที่ 1: FTP Server
  ftp-server:
    image: fauria/vsftpd
    container_name: my-ftp-server-compose
    ports:
      - "21:21"
      - "21100-21110:21100-21110"
    volumes:
      - ./ftp-data:/home/vsftpd/testuser
    environment:
      - FTP_USER=testuser
      - FTP_PASS=testpass
      - PASV_ADDRESS=127.0.0.1

  # Service ที่ 2: FTP Client (โปรแกรมของเรา)
  ftp-client:
    build: . # สั่งให้ build จาก Dockerfile ในโฟลเดอร์ปัจจุบัน
    container_name: my-ftp-client-compose
    depends_on: # กำหนดให้เริ่มทำงานหลังจาก ftp-server พร้อมแล้ว
      - ftp-server
```

คำอธิบายไฟล์ docker-compose.yml:

- **services:** กำหนดชุดของ Container ที่จะทำงานร่วมกัน
- **ftp-server:** คือบริการแรกของเรา ใช้ Image fauria/vsftpd มีการตั้งค่า Port, Volume (สำหรับเก็บไฟล์), และ Environment Variables (สำหรับสร้าง User/Password) เหมือนกับคำสั่ง docker run
- **ftp-client:** คือบริการที่สองซึ่งเป็นโปรแกรมของเราเอง
 - **build:** :: สั่งให้ Docker Compose สร้าง Image จาก Dockerfile ที่อยู่ในโปรเจกต์
 - **depends_on:** เป็นการกำหนดเงื่อนไขที่สำคัญมาก โดยบอกว่า ftp-client จะยังไม่เริ่มทำงานจนกว่า ftp-server จะทำงานเสร็จสิ้นก่อน เพื่อให้แน่ใจว่าเมื่อ Client เริ่มทำงาน Server จะพร้อมรับการเชื่อมต่อแล้ว

การใช้งาน

เราสามารถสร้างและรันแอปพลิเคชันทั้งหมดได้ด้วยคำสั่งเดียวใน Terminal:

`docker-compose up`

และเมื่อต้องการหยุดและลบ Container ทั้งหมด ก็ใช้คำสั่ง:

`docker-compose down`

อ้างอิง

- FluentFTP Official Website. (<https://fluentftp.net/>)
- Docker Official Documentation. (<https://docs.docker.com/>)
- Docker Hub Repository for fauria/vsftpd. (<https://hub.docker.com/r/fauria/vsftpd>)
- Microsoft Corporation. .NET documentation.
(<https://learn.microsoft.com/en-us/dotnet/>)