# 1. 環境設定:檢查 Windows 認證，以及設定本機使用者資訊

10. 檢查 Windows 認證 (Windows作業系統才需操作)

請至「控制台 → 使用者帳戶 → 認證管理員 → Windows 認證 → 一般認證」，

檢查是否已經有 git 或 gitHub 字樣的認證，

如果有，但不是自己的 GitHub 帳號，請先移除。

20. 設定本機使用者資訊。步驟 22~24 請在「命令提示字元 | 終端機」輸入指令

21. 請確認已安裝 Git

22. 查看是否已經有使用者資訊，分別輸入以下指令:

git config --global user.name

git config --global user.email

23. 如有顯示資訊，請確認是否為自己的 GitHub 帳號資訊。

如果不是，請分別輸入以下指令移除:

git config --global --unset user.name

git config --global --unset user.email

24. 設定與 GitHub 帳戶相同的使用者資訊，分別輸入以下指令:

git config --global user.name "你的名稱"

git config --global user.email "你的信箱"

# 2.建立數據庫，並複製(下載)數據庫到本機

10. 建立 GitHub 數據庫

11. 前往 GitHub 官網並登入帳號 (https://github.com/)

12. 在 dashboard 頁面點選綠色「Create repository」或「New」按鈕 建立數據庫

13. 在 Repository name 輸入框，輸入數據庫名稱: Lab\_A

14. 其餘選項維持預設，點擊最下方綠色「Create repository」按鈕建立

20. 複製(下載) GitHub 數據庫到本機。步驟 22~23 請在「命令提示字元 | 終端機」輸入指令

21. 請在桌面建立 Git\_Demo 資料夾，用以存放 Lab\_A 數據庫

22. 請確認 Git\_Demo 資料夾的路徑(右鍵>>>內容>>>C:\Users\User\Desktop)，並使用 cd 指令切換到 Git\_Demo 資料夾

>cd Desktop

23. 使用指令「git clone 數據庫位址」複製(下載) Lab\_A 數據庫

git clone https://github.com/Mateotsai/Lab\_A.git

# 3.「編輯檔案 → 追蹤檔案 → 提交新版本 → 推送到GitHub」的流程

10. 觀察工作目錄目前的狀態

11. 使用 Visual Studio Code 開啟 GitHub 複製下來的 Lab\_A 資料夾

12. 點選上方功能表列 → 終端機 → 新增終端機

13. 從終端機面板右上角的下拉按鈕，選擇 Command Prompt

14. 在終端機輸入 git status 指令，觀察訊息

20. 新增檔案

21. 新增 pokemon.html，輸入以下內容並存檔

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Pokemon</title>

</head>

<body>

<h1>皮卡丘</h1>

</body>

</html>

22. 在終端機輸入 git status 指令，觀察訊息

30. 將檔案放到暫存區列管|追蹤。步驟 31~32 請在終端機操作

31. 輸入 git add pokemon.html

32. 輸入 git status 指令，觀察訊息

40. 提交檔案。步驟 41~42 請在終端機操作

41. 輸入 git commit -m "新增皮卡丘"

42. 輸入 git log，查看版本紀錄

50. 推送到 GitHub 數據庫。步驟 52~54 請在終端機操作

51. 請至 GitHub 頁面確認已登入帳號

52. 輸入 git remote -v，查詢指向的 GitHub 數據庫位址

53. 輸入 git push origin main，若出現 "Connect to GitHub" 視窗，點選 "Sign in with your browser"

54. 輸入 git log --oneline，觀察訊息

55. 在 GitHub 頁面查看數據庫變化

# 4.練習: 建立 Lab\_B 數據庫，並推送第一個版本

請依照下方步驟，在 GitHub 建立新的數據庫，並將 w3schools\_template 資料夾內的檔案推送上去

10. 請在 GitHub 頁面建立新的數據庫 Lab\_B

20. 開啟 命令提示字元，使用 git clone 指令將數據庫複製到 桌面的 Git\_Demo 資料夾裡面

30. w3schools\_template 資料夾內的檔案手動放到 Lab\_B 資料夾 準備推送

40. 使用 Visual Studio Code 開啟 Lab\_B 資料夾

50. 將檔案放到暫存區列管

60. 提交並查看版本紀錄

70. 推送到 GitHub 數據庫

80. 在 GitHub 頁面查看數據庫變化

# 5.編輯檔案，推送新版本(第二版)

10. 編輯檔案

11. 請在 travel.html 加上一個固定在 "左上" 的區塊，如同 "30\_新增元素v1\_結果圖"

12. 想嘗試寫程式的同學，請忽略步驟 13~14

13. 請在 travel.html 加入以下內容

<div class="tyContainer tyLeftSide">

<div class="tyTitle">快速查詢</div>

<a href="#">

<div>機票資訊</div>

</a>

<a href="#">

<div>訂房資訊</div>

</a>

<a href="#">

<div>租車資訊</div>

</a>

</div>

14. 請在 css/style.css 加入以下樣式

.tyContainer {

border: 2px solid white;

width: 100px;

position: fixed;

z-index: 1;

}

.tyLeftSide {

left: 0;

top: 10vh;

}

.tyContainer div {

border: 1px solid white;

color: white;

background-color: black;

width: 100%;

height: 80px;

display: flex;

justify-content: center;

align-items: center;

}

.tyContainer a div:hover {

background-color: #f44336;

}

.tyContainer a {

text-decoration: none;

}

.tyContainer .tyTitle {

background-color: gray;

height: 50px;

}

20. 將所有異動的檔案放到暫存區列管

30. 在本機比較檔案

31. 在終端機輸入 git diff --staged 指令，比較暫存區和目前版本的差異

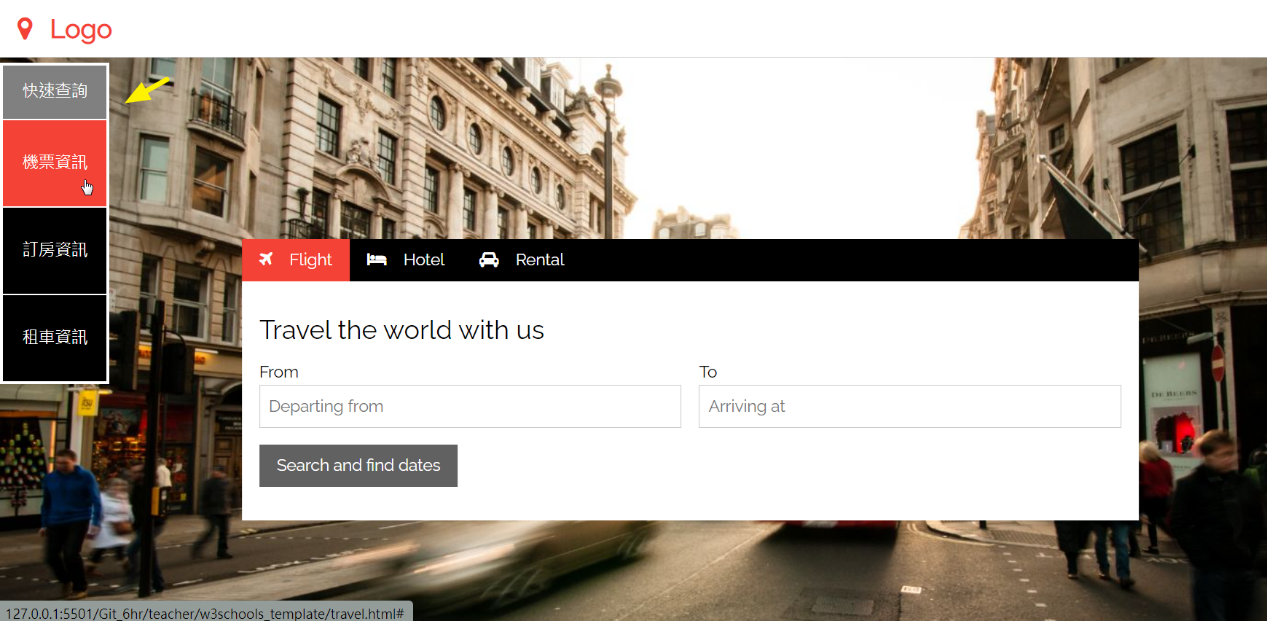
-----------(看完之後按”Q”離開)---------

32. 在 Visual Studio Code 左側的 "Activity Bar" 點選 "原始檔控制"，找到放入暫存區的檔案，點選檔案比較內容差異

40. 提交檔案並查看版本紀錄

50. 在終端機輸入 git push，推送到 GitHub 數據庫

60. 在 GitHub 頁面查看數據庫變化



# 6.捨棄修改:回到第一版的內容 (捨棄第二版修改的內容)

10. 查看版本紀錄

11. 接續 "30\_新增元素v1" 的結果

12. 在終端機輸入 git log --oneline

13. 記下目前版本的雜湊值

20. 回到上一個版本，步驟 21~26 請在終端機操作

21. 輸入「git revert -n 目前版本的雜湊值」，觀察檔案變化 6b66333

22. 輸入 git status，觀察狀態

23. 輸入 git diff --staged，比較檔案

24. 輸入 git commit -m "新增左上區塊(捨棄)"，提交檔案

25. 輸入 git log --oneline，查看紀錄

26. 輸入 git push，推送到 GitHub 數據庫

27. 在 GitHub 頁面查看數據庫變化

# 7.建立新的分支 cat，在 cat 開發新功能並提交新的版本

10. 建立分支

11. 在終端機輸入 git branch cat，建立 cat 分支

12. 輸入 git branch，查看 cat 分支是否建立成功，以及查看目前所在分支

13. 輸入 git checkout cat，切換到 cat 分支

14. 輸入 git branch，查看目前所在分支

20. 編輯檔案

21. 在 travel.html 頁面中間加上一個彈出的視窗，如同 "40\_新增元素v2\_結果圖"

22. 想嘗試寫程式的同學，請忽略步驟 23~25。以下程式從 w3schools 複製並改寫

https://www.w3schools.com/howto/tryit.asp?filename=tryhow\_css\_modal

23. 請在 travel.html 加入以下內容

<div id="tyModal">

<div class="tyModalContent">

<span class="tyClose">&times;</span>

<h1>好康報報</h1>

<p>優惠資訊</p>

</div>

</div>

24. 請在 css/style.css 加入以下樣式

#tyModal {

display: none;

position: fixed;

z-index: 1;

padding-top: 100px;

left: 0;

top: 0;

width: 100%;

height: 100%;

overflow: auto;

background-color: rgba(0, 0, 0, 0.4);

}

.tyModalContent {

background-color: #fefefe;

margin: auto;

padding: 20px;

border: 1px solid #888;

width: 80%;

}

.tyClose {

color: #aaaaaa;

float: right;

font-size: 28px;

font-weight: bold;

}

.tyClose:hover,

.tyClose:focus {

color: #000;

cursor: pointer;

}

25. 請在 js/myScript.js 加入以下 JavaScript

var modalElem = document.getElementById("tyModal");

var spanElem = document.getElementsByClassName("tyClose")[0];

spanElem.onclick = function () {

modalElem.style.display = "none";

}

setTimeout(function () {

modalElem.style.display = "block";

}, 1000);

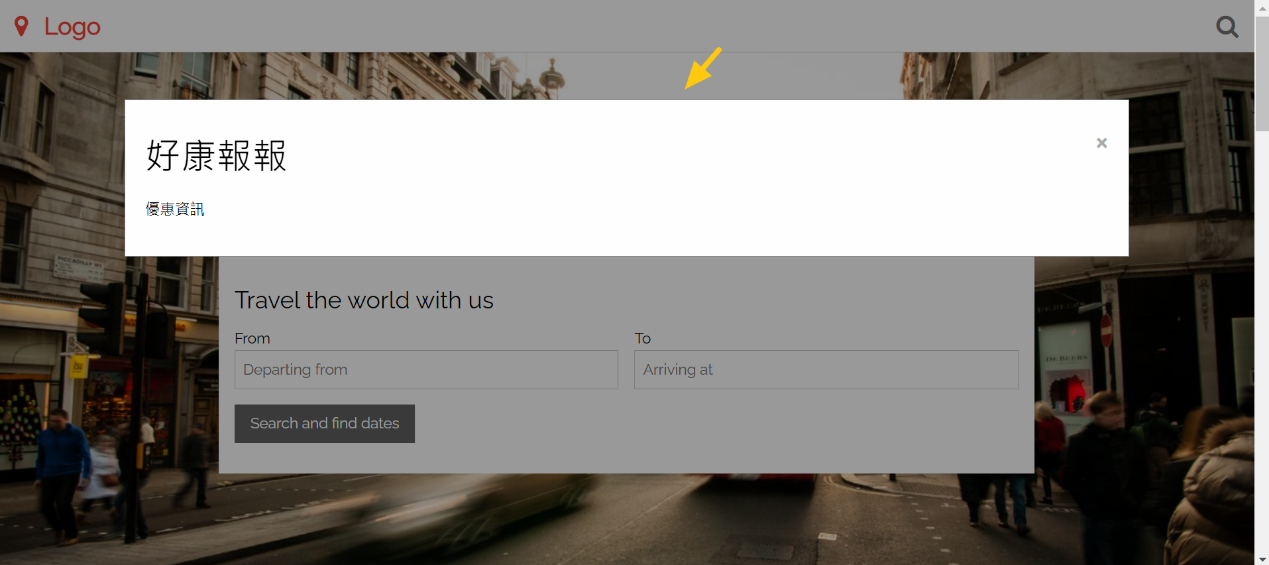
30. 將所有異動的檔案放到暫存區列管

40. 在本機比較檔案

41. 在終端機輸入 git diff --staged 指令，比較暫存區和最近一次提交版本的差異

42. 在 Visual Studio Code 左側的 "Activity Bar" 點選 "原始檔控制"，找到放入暫存區的檔案，點選檔案比較內容差異

50. 提交檔案並查看版本紀錄



# 8.合併分支 : 將 cat 分支合併回 main 分支

10. 切換分支

11. 接續 "40\_新增元素v2" 的結果

12. 在終端機輸入 git checkout main，切換回 main 分支

13. 輸入 git branch，查看目前所在分支

20. 合併分支

21. 在終端機輸入 git merge cat

22. 輸入 git status，查看訊息

23. 輸入 git log --oneline，查看紀錄

30. 在終端機輸入 git push，推送到 GitHub 數據庫

40. 在 GitHub 頁面查看數據庫變化

# 9.提取新版本 : 使用 git pull 指令提取新版本

10. 請在 GitHub 開啟 Lab\_B 數據庫頁面

11. 請先確認左上方的 "Switch branches/tags" 選單 已選取 main

12. 點選右上方 "Add file" 選單，選擇 "Create new file" 按鈕 新增檔案

13. 在上方 "Name your file..." 輸入框，輸入檔案名稱 apple.txt

14. 在下方編輯框輸入以下內容:

"皮卡丘 => 這是在 GitHub 頁面編輯的內容，模擬在教室電腦編輯檔案"

15. 在右上點選 "Commit changes..." 按鈕

16. 輸入提交訊息，點選 "Commit directly to the main branch"，點選右下角 "Commit changes"

20. 在 Visual Studio Code 開啟 Lab\_B 資料夾

21. 在終端機輸入 git branch 查看目前所在分支，若不在 main，請輸入指令 git checkout main 切換分支

22. 輸入 git pull 指令，提取 GitHub 上的檔案

23. 輸入 git log --oneline，查看紀錄

# 10.不同檔案版本衝突 GitHub 及 本機皆有新版本的情況下使用 git pull 指令

10. 請在 GitHub 開啟 Lab\_B 數據庫頁面

11. 請先確認左上方的 "Switch branches/tags" 選單 已選取 main

12. 點選 apple.txt，在編輯框右上點選 "Edit this file" (筆的圖示)

13. 在下方編輯框輸入以下內容:

"伊布 => 這是在 GitHub 頁面編輯的內容，模擬在教室電腦編輯檔案

14. 在右上點選 "Commit changes..." 按鈕

15. 輸入提交訊息，點選 "Commit directly to the main branch"，點選右下角 "Commit changes"

20. 在 Visual Studio Code 開啟 Lab\_B 資料夾

21. 在終端機輸入 git branch 查看目前所在分支，若不在 main，請輸入指令 git checkout main 切換分支

22. 新增檔案 bee.txt，輸入以下內容:

"小火龍 => 這裡是在 VS Code 編輯的內容，是在家裡電腦編輯的檔案"

23. 將檔案放到暫存區列管

24. 提交檔案並查看版本紀錄

25. 輸入 git push，推送到 GitHub 數據庫

26. 推送失敗，請輸入 git pull 指令，先提取 GitHub 上的檔案

進入vim文字編輯器

退出 :QA

編輯 I 🡺 編輯完按 ESC 🡺 :WQA +(Enter)

27. 再次輸入 git push

28. 在 GitHub 頁面查看數據庫變化

# 11.同檔案版本衝突 解決 git pull 之後的衝突

10. 請在 GitHub 開啟 Lab\_B 數據庫頁面

11. 請先確認左上方的 "Switch branches/tags" 選單 已選取 main

12. 點選 apple.txt，在編輯框右上點選 "Edit this file" (筆的圖示)

13. 在下方編輯框輸入以下內容:

"妙蛙種子 => 這是在 GitHub 頁面編輯的內容，模擬在教室電腦編輯檔案"

14. 在右上點選 "Commit changes..." 按鈕

15. 輸入提交訊息，點選 "Commit directly to the main branch"，點選右下角 "Commit changes"

20. 在 Visual Studio Code 開啟 Lab\_B 資料夾

21. 在終端機輸入 git branch 查看目前所在分支，若不在 main，請輸入指令 git checkout main 切換分支

22. 在 apple.txt 輸入以下內容:

"胖丁 => 這裡是在 VS Code 編輯的內容，是在家裡電腦編輯的檔案"

23. 將檔案放到暫存區列管，以及提交檔案並查看版本紀錄

🡺 git add apple.txt

🡺 git commit -m “新增版本胖丁”

🡺 git log --oneline

24. 輸入 git push，推送到 GitHub 數據庫 🡺錯誤訊息

25. 推送失敗，請輸入 git pull 指令，提取 GitHub 上的檔案

26. 出現衝突，在編輯畫面右下角 選擇 "在合併編輯器中解析"

27. 解決衝突之後，點選右下角 "完成合併"

28. 提交合併結果

29. 輸入 git push，推送到 GitHub 數據庫，在 GitHub 頁面查看數據庫變化

# Git指令

**※執行指令之前，請注意當下所在的目錄**

| 用途 | 指令 | 範例 |
| --- | --- | --- |
| 查看 Git 版本 | git version |  |
| 查看使用者名稱和信箱 | git config --global user.name  git config --global user.email |  |
| 清除使用者名稱和信箱 | git config --global --unset user.name  git config --global --unset user.email |  |
| 設定使用者名稱和信箱 | git config --global user.name "你的名稱"  git config --global user.email "你的信箱"  ＊建議使用者名稱和信箱與 GitHub 帳戶一致 | git config --global user.name "luffy123"  git config --global user.email "luffy123@gmail.com" |
| 複製(下載) GitHub 數據庫至本機 | git clone 數據庫位址  ＊適用 : 本機尚未存在這個數據庫 (通常只會執行一次) | git clone https://github.com/luffy123/Lab\_A.git |
| 查看工作目錄及暫存區的狀態 | git status |  |
| 將指定檔案放到暫存區列管(追蹤) | git add 檔案名稱 | git add pokemon.html |
| 將特定類型的檔案放到暫存區列管(追蹤) | git add \*.副檔名 | git add \*.html |
| 將所在目錄的全部檔案放到暫存區列管(追蹤) | git add . |  |
| 將暫存區的檔案提交到本機倉庫(提交新版本) | git commit -m "提交訊息" | git commit -m "新增第一版" |
| 將工作目錄的檔案直接提交到本機倉庫(提交新版本) | git commit -a -m "提交訊息"  ＊新增的檔案無法使用此指令 | git commit -a -m "新增第二版" |
| 查看版本紀錄 | git log |  |
| 查看版本紀錄，每筆紀錄以單行顯示 | git log --oneline |  |
| 將檔案推送(上傳)到 GitHub 數據庫 | git push 數據庫位址  分支名稱git push | git push https://github.com/luffy123/Lab\_A.git main  git push origin main |
| 查看 origin 指向的數據庫位址 | git remote -v |  |
| 設定 GitHub 數據庫的簡稱 | git remote add  自訂的簡稱 數據庫位址  ＊若 git remote -v 沒有資料，才需要執行以上指令 | git remote add apple https://github.com/luffy123/Lab\_A.git |
| 比較工作目錄和暫存區的差異 | git diff |  |
| 比較暫存區和目前版本的差異 | git diff --cached  git diff --staged |  |
| 捨棄指定版本(不會自動提交) | git revert -n 指定版本的雜湊值 | git revert -n 2a3b4cd |
| 捨棄指定版本(會自動提交) | git revert 指定版本的雜湊值 | git revert 2a3b4cd |
| 查看分支 | git branch |  |
| 建立分支 | git branch 分支名稱 | git branch cat |
| 切換分支 | git checkout 分支名稱 | git checkout cat |
| 建立分支並切換 | git checkout -b 分支名稱 | git checkout -b cat |
| 刪除分支 | git branch -d 分支名稱 | git branch -d cat |
| 合併分支 | git merge 分支名稱 | git merge cat |
| 提取 GitHub 數據庫檔案  (取得新版本) | git pull  ＊適用 : 事前已經有複製一份 GitHub 數據庫到本機 |  |