Line LIFF 視訊錄影技術手冊

作者:杜晟愷

版本:1.0.1

1. 系統功能簡介

利用Line LIFF結合手機前視訊鏡頭進行錄影，並將錄下的影像傳送到後台並以mp4影音格式儲存

1. 系統開發工具
2. 前端:

本系統前端使用者介面利用HTML、CSS組成，並以Javascript的Vue JS套件作為邏輯處理的工具

* 1. HTML

作為網頁製作最常使用到的語言，HTML固然是此系統於前端使用者介面實作時的首選語言，它讓我們可以快速製作出需要的樣板

* 1. CSS

利用CSS做出更美觀的前端視覺效果，其中本系統多套用了以下CSS套件

* + 1. Bootstrap

版本: 5.0.2

透過使用Bootstrap可以讓系統在實作CSS的美化時更方便且有RWD的效果

* 1. Javascript

本系統於實作時也使用了以下套件，將處理邏輯做到更完善

* + 1. Vue JS\*

版本: 2.6.14

Vue JS 作為當今最熱門的JS套件是我們主要的首選，它讓我們再實作前端實可以以更快速與簡單的邏輯完成

* + 1. Axios

版本: 0.9.1

雖然Javascript有內建的fetch api，但相比之下axios在撰寫實較為直覺加上優點又比較多，故在本系統中，所有request相關的實作都是以axios完成

* + 1. Jquery

版本: 3.6.0

著名的套件，在本系統中主要當作選擇器使用

本次使用的js的套件都是以CDN的形式載入，可以大幅降低載入時間以及系統大小，並也有比較高的安全性與可靠度

1. 後端

後端使用語言以Javascript為主，並結合廣為人知的express js實作後端，以下為使用到的套件:

* 1. express js

版本: 4.17.1

Express js 是一款廣為人知的框架，本系統利用express js實作所需的api

* 1. fluent-ffmpeg

版本: 2.1.2

在使用fluent-ffmpeg之前，請先確認是否已將ffmpeg加入path中\*\*

此套件的用處在於將使用者錄影的資訊由webm格式轉為mp4格式，以提供後續影像辨識實使用

* 1. fs

node js內建的功能，讓我們可以實作檔案的增刪與管理

* 1. body-parser

版本: 1.4.5

由express js開發團隊開發的套件，用來剖析request的body

* 1. stream-buffers

版本: 3.0.2

一款將buffer物件轉為streambuffer的知名套件，可以在後續轉檔時不須先創建一個多餘的檔案並刪除即可透過fluent-ffmpeg轉為mp4

* 1. ejs

版本: 3.1.6

用於動態渲染前端的html檔，將csrf-token傳給前端已達到進行Cross-site request forgery的初步防護

* 1. helmet

版本: 4.6.0

由express官方推薦使用的最基礎資安防護中介層，包含11大項的資安防護

其中的contentSecurityPolicy之script-src與style-src為載入CDN與本系統本身包含的js檔，有做客製化的參數，如有需要可再進行修改

* 1. csurf

版本: 1.11.0

同樣由express團隊開發的中介層套件之一，用於產生csrf-token給前端並執行防護

* 1. cookie-parser

版本: 1.4.5

同樣由express團隊開發的中介層套件之一，搭配csrf使用已達到防護Cross-site request forgery

* 1. child\_process

node js 中包含的套件，可以呼叫一個子行程處理要做的命令。

本系統中若沒有事先將ffmpeg加入倒Path中的話會啟用這項程序，利用包含在檔案內部的ffmpeg執行檔以命令列的方式執行轉檔，但效率會比使用path的ffmpeg差。

1. 系統執行方式

如要執行本系統，請依照以下步驟進行

* 1. 於系統檔案目錄下執行npm install

他會將需要的node js package進行安裝

* 1. 將ffmpeg加入path中

Ubuntu系統:

先執行sudo apt-get update && sudo apt-get upgade後

執行sudo apt-get install ffmpeg即可

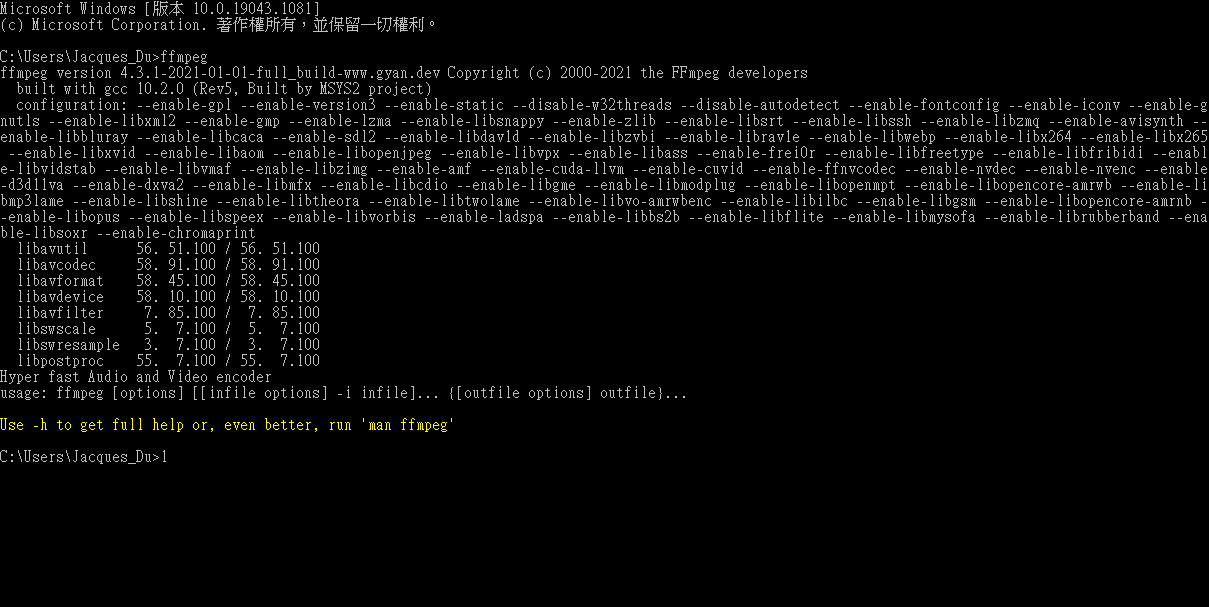
Windows系統:

於ffmpeg官網(https://www.ffmpeg.org/)下載ffmpeg最新版的檔案，於環境變數的path新增變數值為path/to/ffmpeg.exe，如下圖所示



兩個系統測試ffmpeg是否已成功加入path中的方法為:

在terminal/cmd中直接輸入ffmpeg，應該要出現類似下圖內容



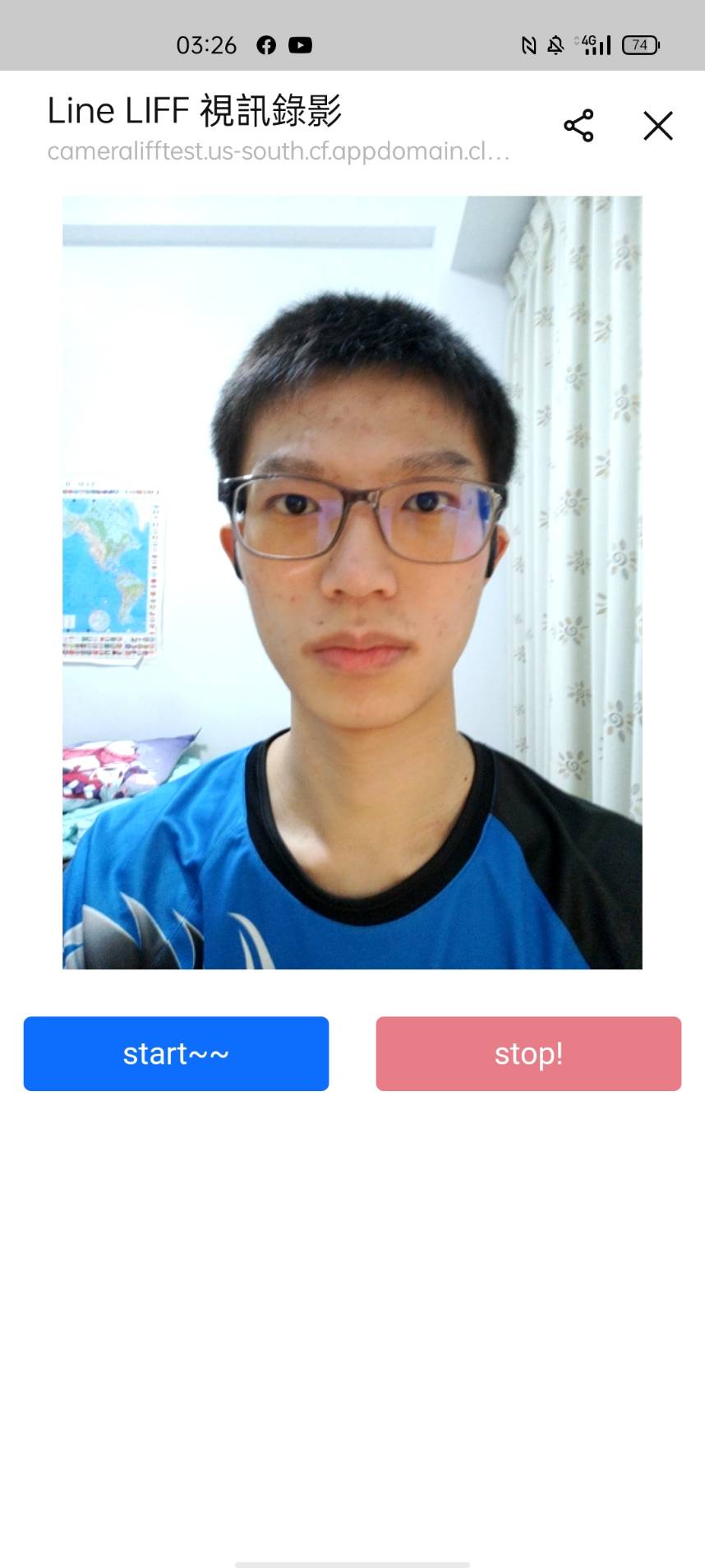
* 1. 執行npm start

本系統預設的port 為8080，後續可將port的值直接修改或是在系統變數中加入port，並設定需要的值

* 1. 於瀏覽器中進入<http://localhost:YourPort>

進入之後應可看到下圖(首次進入須先允許使用相機權限)

會看到使用者介面如下圖



使用者於按下「start~~」後會開始錄影，此時的stop!將會變為enable。

於錄影完成後按下stop!可將錄影下來的影像post一個request給/postStream給後台，後台會於渲染完影片後將使用者介面恢復為可以錄影狀態。

* 1. 如需要觀看影片是否有成功錄影，除了可以直接於./backend/videos/資料夾中找到影片外，亦可至http://localhost:8080/backend可以呈現目前檔案資料夾中的影片數量，如下圖:



1. 備註

\* 未來將會改用React JS進行改寫

\*\* 若無法正常執行將ffmpeg依舊是可以讓系統運作，但是效率較差因此不建議

如有任何問題請聯絡:

杜晟愷

電話: 0928823161

電子信箱: justinduthebest@gmail.com