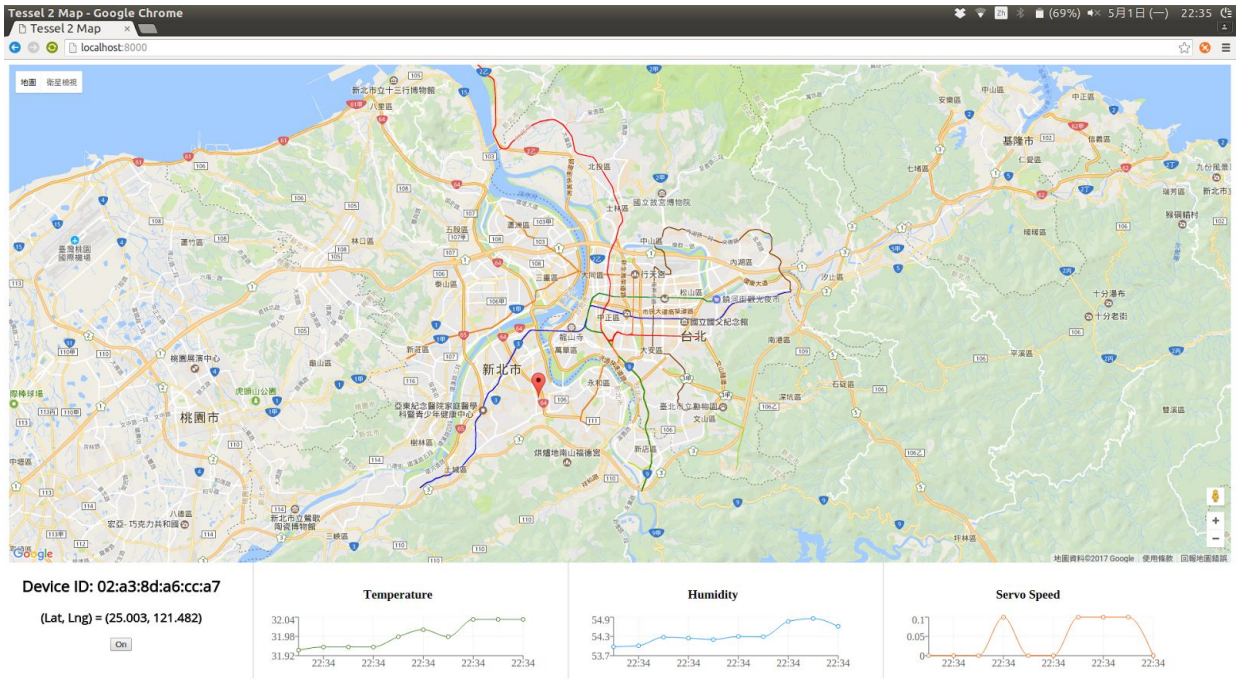
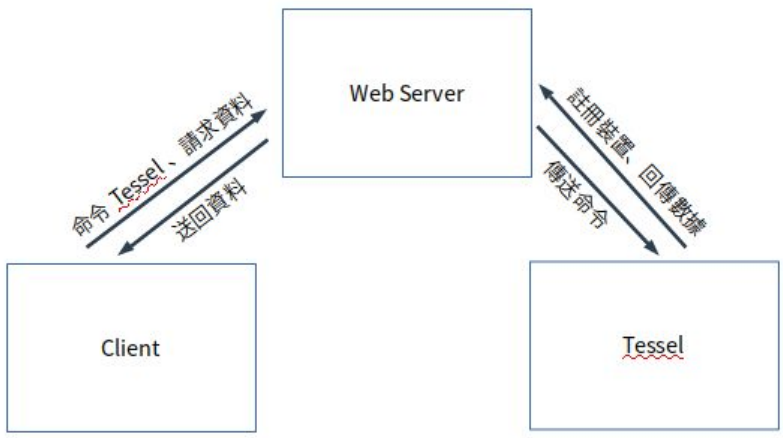
Embedded System Lab Assignment 2

b02901032 張桓誠　　b02901068 林楷恩



這次作業我們要做的是一個利用IOT以及Web Programming的智慧溫溼度控制器，利用Tessel 2提供的溫溼度感應器，將資料傳到伺服器後，使用者可以利用類似Google Map的介面，點擊並查看感應器所在位置的溫溼度趨勢圖，並且依此決定是否要啟動馬達，模擬開動排風扇的功能。



Front-end

前端的部分，我們主要還是使用ReactJS當作開發的主要套件，搭配和Google Map API結合的react-google-map，以及整合React跟d3.js的Recharts，來達成前述的功能。以下就介紹我們所使用到較為特別或是重要的幾個要點，並且簡單描述我們學到了甚麼：

- fetch

  介紹：

  為了要能夠利用JavaScript去送出http request，我們使用了fetch API。它能夠讓我們預先決定好要送出的request，並且可以即時決定要傳出的資料是甚麼。例如依據使用者點擊地圖上的感應器來送出相對應的感應器ID，以取得其歷史資料。

  遭遇到的困難：

  這部分遇到最大的困難是CORS的設定，CORS最主要的目的是防止惡意網站利用使用者的連線去攻擊其他網域的網站資源。但也因為這樣，我們想要跨越兩個不同網域去抓取API時，就會得到錯誤訊息。也為此研究了相當久的時間，最後發現CORS的設定是必須要在伺服器端回傳允許的標頭，這樣使用者端才能夠成功接收。

- ReactJS

  介紹：

  由Facebook開發的一款套件，我們認為最方便的地方是它的寫法，因為相當類似以前C++學到的物件導向的方式，比較好管理網站中複雜的元素，而且也能夠避免使用到太多全域變數。

- react-google-map

  介紹：

  我們的網站需要一個Google Map令使用者可以清楚知道自己的感應器所在的位置，因此選用了這個套件。他最大的功用就是將Google Map整合成React的JSX寫法，所以在使用上相當直覺，不必再自己想辦法將Google Map API嵌入，而是直接使用。另外他也將marker和map拆開，而且只要將要求的參數以props的方式傳入就會自動在地圖上標點，相當方便。

- Recharts

  介紹：

  一開始是打算使用d3.js，但仔細研究之後發現太過複雜，並且還要再自行把SVG的rendering和React整合在一起，因此就再找了這個套件包。它的優點就是把上述的問題解決並且也是提供JSX讓使用者操作，在deploy之後它會自行展開成SVG的元素，並且把該有的部分也一併處理好。此外，它內建就有相當美觀的動畫以及點擊時會產生詳細資訊，就不需要自己再處理複雜的邏輯。

Back-end

- Web server

  使用 restify，有以下 API：
  - GET /api/realTime: 拿所有 device 的座標和最新溫溼度
  - POST /api/histData: 傳 deviceId 進來，拿特定 device 的即時和歷史資料
  - PUT /api/control: client 透過這個對 Tessel 下指令
  - POST /api/device: Tessel 透過這個傳資料到 web server
  - POST /api/register: Tessel 先透過這個註冊 id 和座標後才能傳資料

- Tessel

  使用 climate 和 servo 模組，並且在上面開了個 server 以和 web server 溝通。server 只有開一個 API Endpoint，/command，拿來處理傳過來的指令。
  和 web server 註冊後，會定期跟 server 回傳現在 climate 讀到的值以及 servo 的轉動情形。
  因為只有 servo，所以現在只處理 servo 的相關指令，可以讓他轉動或停止。

  限制：
  Tessel 1 有個 Hardware ID 的 API 可以用，但 Tessel 2 被鎖起來了，所以後來使用 MAC Address 當作 deviceId。因為 Tessel 可以用的 port 只有兩個，就不能用原本計畫的 GPS 模組，所以現在座標是寫死的，另外因為只是測試的關係，web server 的位置會隨 Wifi AP 不同而變化，正式使用的時候應該可以用 domain name 解決，但座標仍然沒辦法。
  還有 servo 那塊板子實在太慘啦！曾經出現那麼一下下又不見了，所以後面都用模擬機跑，假裝有東西。