臺北科技大學資訊工程系

110 學年度實務專題計畫成果報告

基於質量彈簧模型
之無線感測網路定位系統

專題編號：110-CSIE-S012

專題計劃參與人員：107590034　鄧鎧晨

107590035　許哲維

107830018　江品寬

指導教授：吳和庭 教授

執行期間：109 年 1 學期至 110 年 1 學期

章節目錄

一、 背景

這些年，無線相關產品蓬勃的發展，就連家庭的設備也趨近於無線化的趨勢。無線感測網路(Wireless Sensor Network)是一種結合感測、運算及網路能力的技術，由一到多數個無線資料接收器所構成的網路，將每個感測器串聯成感測網路，利用感測器的感應範圍，偵測週遭環境及特定目標，將一些的簡易資料做運算的動作之後，回傳到資料接收器，進而讓我們方便擷取資料，並進一步透過所獲得資料來瞭解環境的狀況，透過這些資料來做調整、維護或提供更好的服務。為了達到接收器大量的佈置，必須具備低成本、低功耗、體積小、容易佈建，並具有感測環境裝置、可程式化控制、可動態組成的特性。

而在滿足工業物聯網的無線感測環境下，實現高準確度的定位演算法，為了達成此目的，必須先合乎工業物聯網的環境，但因全球定位系統(GPS)無法進行室內定位，故我們希望透過 RSSI 測距與質量彈簧模型製作出一個能夠室內定位的系統，並且透過網頁來呈現出感測點的位置與資訊。

資料視覺化前難以判斷資訊　　資料視覺化後資訊清楚明瞭

（圖一）差異示意圖

二、 研究動機與目的

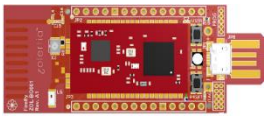
無線感測網路定位系統是近年來熱門的應用領域，而本專題以質量彈簧模型之無線感測網路定位演算法為基礎，利用 CC2538 開發板與 Contiki OS 來架設工業物聯網之室內定位模型，然而該如何以視覺化方式呈現感測點的位置與資訊室本專題著墨的重點。

因此，我們藉由 JavaScript、MongoDB 等等為基礎來架設系統平台，並將此平台架設於實驗空間中；另外，在定位系統中加入地圖資訊、定位點座標，使負責人可隨時透過此系統快速了解系統內感測點之所在位置、狀況等資訊，並將傳送來的資料整理、並進行資料視覺化後，達成使用者友善之系統介面，使資訊更為清楚明瞭。

關鍵詞：質量彈簧模型、無線感測、網路定位系統、Conitki OS、MongoDB、JavaScript。

三、 開發工具與環境

(一)、CC2538 開發板



CC2538 開發板為一個可應用 IEEE 802.15.4 和 ZigBee 之無線控制晶片開發板，具有低功率消耗、安全硬體加速等等特點，適用於物聯網、無線感測網路、智能家電等等。

(二)、Contiki OS

Contiki OS 是一個小型、開源、極易移植的多工電腦作業系統。用於一系列的記憶體受限的網路系統，只需幾千位元組的代碼和幾百位元組的記憶體就能提供多工環境和內建 TCP/IP 支援，藉此實現質量彈簧模型的室內定位系統。

(三)、Node.js

Node.js 是能夠在伺服器端運行 JavaScript 的開放原始碼、跨平台執行環境，其採用了 Google 開發的 V8 執行程式碼，使用事件驅動、非阻塞和非同步輸入輸出模型等技術來提高效能，可優化應用程式的傳輸量和規模。這些技術通常用於資料密集的即時應用程式。

四、 相關技術

(一)、C

是一種通用的程式語言，廣泛用於系統軟體與應用軟體的開發。本專題之 Contiki 端程式皆以 C 作為基礎。

(二)、MongoDB

MongoDB 是一個基於分散式檔案儲存的資料庫，為網路應用提供可擴充套件的高效能資料儲存解決方案。在本專題用來儲存感測點的位置及其更多的資訊。

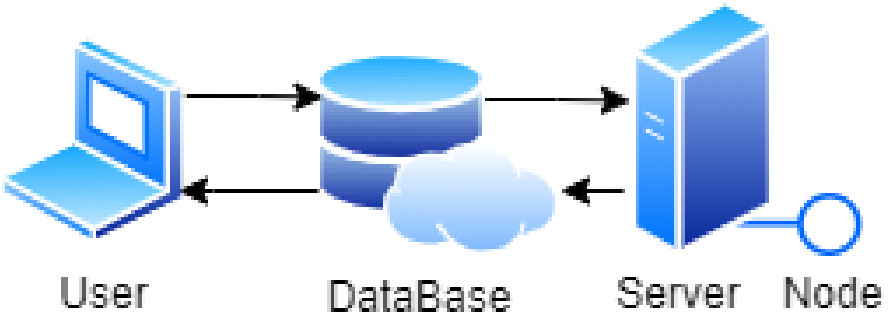
(三)、Express.js

Express.js 是一個簡潔靈活的 Node.js Web 應用框架，提供了一系列強大特性創建各種 Web 應用，和豐富的 HTTP 工具，可快速架設一個完整的網站。

(四)、Chart.js

Chart.js 是用於數據可視化的免費開源 JavaScript 庫，它支持 8 種圖表類型：條形圖，線條圖，面積圖，圓餅圖，氣泡圖，雷達圖，極坐標圖和散點圖。

五、 架構



（圖二）系統架構圖

(一)、前端網站(User)

將資料庫儲存的資料，傳送到定位系統之前端網頁，並以表格與座標圖等方式呈現資料內容。

(二)、資料庫(DataBase)

儲存接收點收到之傳送資訊，並儲存於本機端之 MongoDB，且當前端要求資料時將資料傳送過去。

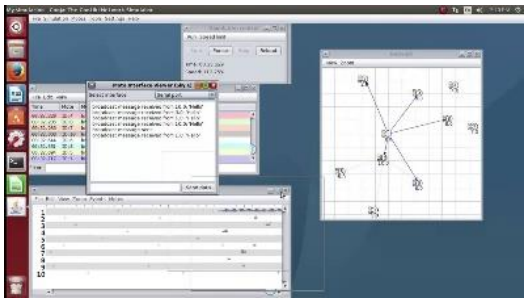
(三)、接收點(Server & Node)

接收點會接收其他 node 傳送而來的資訊(以.json 方式)，並將其資訊儲存於資料庫。

六、 開發流程

(一)、流程設計、Cooja 模擬

為了確保實體 CC2538 開發板可順利運作，在實驗前期先以 Contiki 內建之 Cooja 來模擬實驗結果，並以資料接收為優先研究事項。



（圖三）Cooja 實驗圖

(二)、系統架構設想

在專題開發後期，評估定位系統所需之功能、架構等等，並與演算法組整合傳輸的封包格式、接收模式。

(三)、實驗流程

專題「無線感測網路定位系統」的實驗操作流程如下，分別是程式執行與網頁呈現。

- 先執行 Client、Server、Boarder Router
  1. **執行 Web Server**(此時 Web Server 會把資料從 Boarder Router 的 IPv6 address 抓下並傳到 Database)

     node server.js
  2. **執行後端**

     node server.js

     npm run serve
- 開啟網頁，網頁提供三種呈現方式：
  1. 透過**搜尋定點 IPv6 address** 可得其位置資訊
  2. 透過 **Node List** 中選出想要查看的點即可顯示其位置資訊
  3. 最下方**圖示化呈現**所有的點，游移到點上即可查看詳細資訊

七、 實驗結果

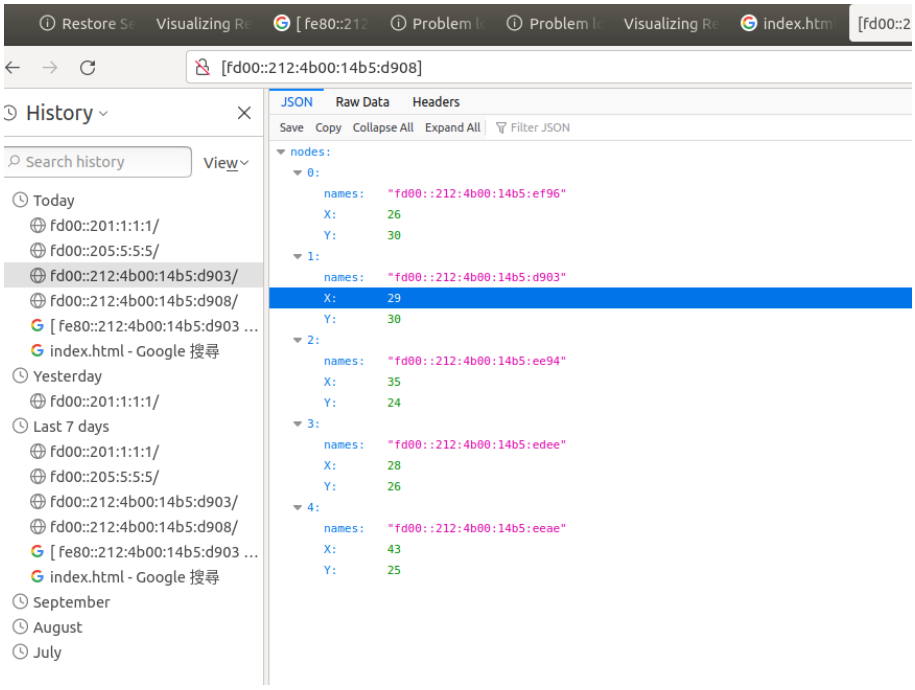
(一)、系統操作畫面

- 燒錄(圖四)

```
t1@ubuntu:~/contiki-ng/examples/rpl-border-router-1$ make TARGET=zoul BOARD=fire
fly PORT=/dev/ttyUSB0 border-router.upload
  CC        ../../arch/cpu/cc2538/./ieee-addr.c
  CC        webserver/webserver.c
  CC        ../../arch/cpu/cc2538/cc2538.lds
  CC        ../../arch/cpu/cc2538/./startup-gcc.c
  CC        border-router.c
  LD        build/zoul/firefly/border-router.elf
  OBJCOPY   build/zoul/firefly/border-router.elf --> build/zoul/firefly/border-r
outer.bin
Flashing /dev/ttyUSB0
Opening port /dev/ttyUSB0, baud 460800
Reading data from build/zoul/firefly/border-router.bin
Cannot auto-detect firmware filetype: Assuming .bin
Connecting to target...
CC2538 PG2.0: 512KB Flash, 32KB SRAM, CCFG at 0x0027FFD4
Primary IEEE Address: 00:12:4B:00:14:B5:D9:08
    Performing mass erase
Erasing 524288 bytes starting at address 0x00200000
    Erase done
Writing 516096 bytes starting at address 0x00202000
Write 8 bytes at 0x0027FFF8F00
```

- 傳送至資料庫(圖五)

```
connect

1
1 document updated
0
1 document updated
[
  {
    _id: new ObjectId("616cbe12366be2eb7d4188f7"),
    names: 'fd00::212:4b00:14b5:eeae',
    X: 31,
    Y: 33
  }
]
false
[
  {
    _id: new ObjectId("616cbdf5366be2eb7d4188f6"),
    names: 'fd00::212:4b00:14b5:edee',
    X: 31,
    Y: 33
  }
]
false
4
1 document updated
3
1 document updated
```



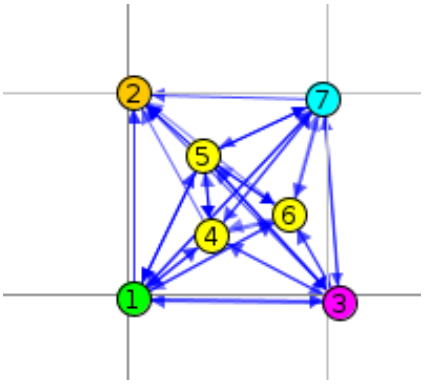
- 前後端運行(圖六)

```
DONE  Compiled successfully in 15511ms


  App running at:
  - Local:   http://localhost:8081/
  - Network: http://10.101.9.167:8081/

  Note that the development build is not optimized.
  To create a production build, run npm run build.
```

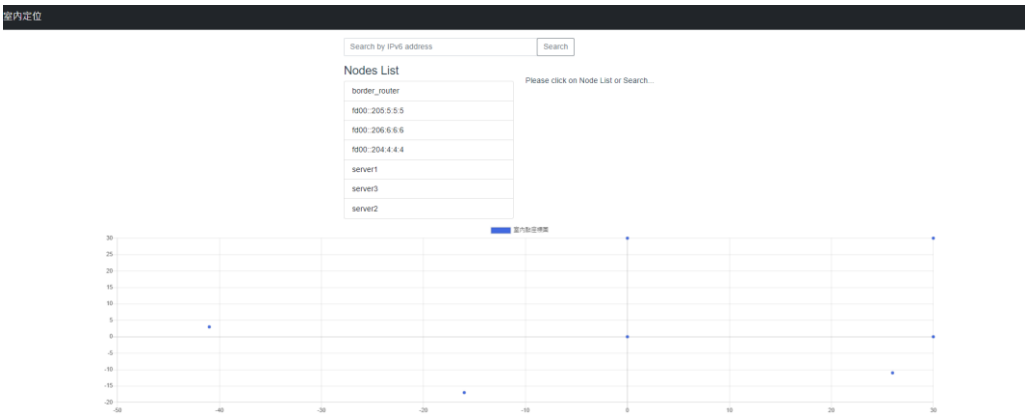
```
PS C:\Users\江品寰\Desktop\special_topic\hello-exprss\full\nodejs-express-mongodb> node .\server.js
Server is running on port 8080.
 http://localhost:8080/
```

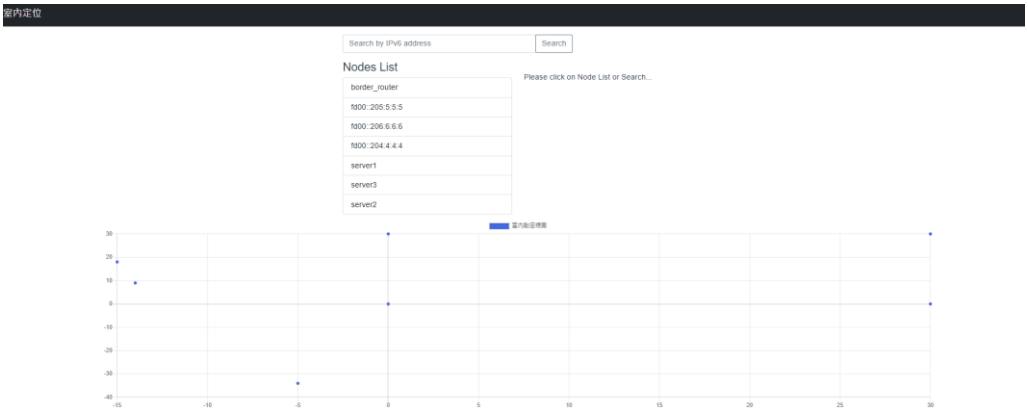
- 座標與傳送狀態(圖七)


```
nodes:
  0:
      names:    "fd00::206:6:6:6"
      X:        26
      Y:        -11
  1:
      names:    "fd00::204:4:4:4"
      X:        -41
      Y:        3
  2:
      names:    "fd00::205:5:5:5"
      X:        -16
      Y:        -17
  3:
      names:    "server1"
      X:        30
      Y:        0
  4:
      names:    "server2"
      X:        0
      Y:        30
  5:
      names:    "server3"
      X:        30
      Y:        30
```

- 前端網頁顯示(圖八)





透過讀取資料庫內之點座標與其他資訊，並在網頁上畫出其相對位置與顯示該點的其他資訊，並且能自動刷新網頁頁面。

(二)、實驗照片

（圖八）實體實驗

八、 問題與解決

- 團隊溝通問題

  因 Covid-19 疫情的關係，可能在分工進度的討論不是那麼的順利。

Sol. 利用 Line、DisCord 等軟體增加討論的時間。

- Contiki 相關資源不足

  因 Border Router 送資訊到資料庫的部分，網路上基本上沒有相關資料，而 Contiking 的使用手冊僅有簡略的說明。

Sol. 自行嘗試解決，如查詢相關論文、修改相關範例程式碼來解決。

- 定位顯示問題

  在前端座標圖顯示出錯誤之點座標。

Sol. 經確認為測試資料，將其資料刪除後即可正常顯示。

九、 結論

專題成功實作出室內定位系統，能夠接收到資料並存於資料庫且能夠正常顯示點座標於座標圖並能自動刷新，雖然在技術、演算法、呈現、與應用等等方面還是有些許的不成熟，但期許未來還可以改進、優化室內定位系統。

十、 未來展望

- 顯示感測資訊(如溫度、濕度等等)
- 感測資訊應用(如高溫警示、訊息通知)
- 前端介面優化

十一、 參考文獻

[1] Xavier Vilajosana, Thomas Watteyne, Tengfei Chang, Mališa Vucini, Pascal Thubert, and Simon Duquennoy, “IETF 6TiSCH: A Tutorial,” IEEE COMMUNICATIONS SURVEYS & TUTORIALS, Vol. 22, No. 1, pp. 595-615, 2020.

[2] 李宜倫，「基於質量彈簧模型之無線感測網路定位演算法」，**國立臺北科技大學資訊工程系碩士班碩士學位論文**，108 年。

[3] Agus Kurniawan, “Practical Contiki-NG”, Apress, 2018

[4] Texas Instruments, “CC2538 适用于 2.4GHz IEEE 802.15.4、6LoWPAN 和 ZigBee®应用的强大无线微控制器片上系统”, https://tinyurl.com/3fsyue68, 2015

[5] Refsnes Data, “W3schools”, Internet: www.w3schools.com, 1999-2020

[6] ARENSKI、吳嘉芳，「圖解 LAYOUT 33 種版面設計圖解，新手也能會！」，**旗標科技股份有限公司**，109 年。

[7] 玉飼真一、村上竜介、佐藤哲、太田文明、常盤作、吳嘉芳，「Web 設計職人必修 UX Design 初學者學習手冊」，**旗標科技股份有限公司**，107 年。

附錄

Contiki

Websense.c

```
#include "contiki.h"
#include "sys/log.h"
#include "net/ipv6/uiplib.h"
#include "net/routing/routing.h"
#include "net/ipv6/uip-ds6-nbr.h"
#include "net/ipv6/uip-ds6-route.h"
#include "net/ipv6/uip-sr.h"
#include "net/ipv6/simple-udp.h"
#include "net/netstack.h"
#include "net/packetbuf.h"
#include "httpd-simple.h"
#include "net/routing/rpl-location/rpl-loc.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define LOG_MODULE "RPL WEB"
#define LOG_LEVEL LOG_LEVEL_INFO

#define WITH_SERVER_REPLY  0
#define UDP_CLIENT_PORT    8765
#define UDP_SERVER_PORT    5678
#define SEND_INTERVAL          (30 * CLOCK_SECOND)

static struct simple_udp_connection udp_connSC;
PROCESS(web_sense_process, "Sense Web Demo");
int x;
int y;
int w;
int in=0;
int num = 0;
char result[30];
char temp[30];
int count = 0;
const uip_ipaddr_t* seat;
/*---------------------------------------------------------------------------*/
static char buf[1024];
static int blen;
#define ADD(...) do {                                                     \
      blen += snprintf(&buf[blen], sizeof(buf) - blen, __VA_ARGS__);      \
   } while(0)
#define SEND(s) do { \
   SEND_STRING(s, buf); \
   blen = 0; \
} while(0);

/* Use simple webserver with only one page for minimum footprint.
 * Multiple connections can result in interleaved tcp segments since
 * a single static buffer is used for all segments.
 */
#include "httpd-simple.h"

/*---------------------------------------------------------------------------*/
//node struct
```

```
struct node_t{
	char address[30];
	int x;
	int y;
};

struct node_t nodes[100];


/*-----------------------------------------------------------------*/

/*-----------------------------------------------------------------*/
///use to exchange type of ipv6 address
void change(char a){
	if(a=='0'){
	 ADD("0");
	}
	else if(a=='1'){
	 ADD("1");
	}
	else if(a=='2'){
	 ADD("2");
	}
	else if(a=='3'){
	 ADD("3");
	}
	else if(a=='4'){
	 ADD("4");
	}
	else if(a=='5'){
	 ADD("5");
	}
	else if(a=='6'){
	 ADD("6");
	}
	else if(a=='7'){
	 ADD("7");
	}
	else if(a=='8'){
	 ADD("8");
	}
	else if(a=='9'){
	 ADD("9");
	}
	else if(a=='a'){
	 ADD("a");
	}
	else if(a=='b'){
	 ADD("b");
	}
	else if(a=='c'){
	 ADD("c");
	}
	else if(a=='d'){
	 ADD("d");
	}
	else if(a=='e'){
	 ADD("e");
```

```
        }
        else if(a=='f'){
          ADD("f");
        }
        else if(a==':'){
          ADD(":");
        }

}
/*---------------------------------------------------------------------------
static void
udp_rx_SC_callback(struct simple_udp_connection *c,
                   const uip_ipaddr_t *sender_addr,
                   uint16_t sender_port,
                   const uip_ipaddr_t *receiver_addr,
                   uint16_t receiver_port,
                   const uint8_t *data,
                   uint16_t datalen)
{
    rpl_loc_msg_t msg;
      msg = *(rpl_loc_msg_t *)data;
      //printf("get msg from ");
    seat = sender_addr;
    printf("\nWeb\n");
    LOG_INFO_6ADDR(sender_addr);
    printf("\n");
    LOG_INFO_6ADDR(seat);
    if(msg.msgType == msg4)
      {
        printf("\nx = %d, y = %d\n", msg.loc_x, msg.loc_y);
        x = msg.loc_x;
        y = msg.loc_y;
        printf("\nWeb2\n");
        uint16_t a;
    int i;
    int j,k;

        for(i=0;i<30;i++){
      result[i] = '\n';
      temp[i] = '\n';
              count=0;
        }

        for(i = 0; i < sizeof(uip_ipaddr_t)/2; i ++) {

        a = (seat->u16[i]);
        sprintf(temp, "%x", a);
        if(i==0){
      result[count++] = temp[2];
      result[count++] = temp[3];
      result[count++] = temp[0];
      result[count++] = temp[1];
        }
        else if(i==1){
      result[count++] = ':';

        }
        else if(i==2){
```

```
result[count++] = ':';
  }
  else if(i==3){

  }
  else if(i==4){

result[count++] = temp[2];
result[count++] = temp[1];
result[count++] = temp[0];

  }
  else if(i==5){
result[count++] = ':';
          //ADD(":");
//printf("\n:");
for(j=2;j>=0;j--){
      if(temp[j]!='0')
    break;
}
for(k=j;k>=0;k--){
          result[count++] = temp[k];
}
  }
  else if(i==6){
result[count++] = ':';
//ADD(":");
//printf("\n:");
for(j=2;j>=0;j--){
      if(temp[j]!='0')
    break;
}
for(k=j;k>=0;k--){
//          printf("\n%c",temp[k]);
    // ADD("%c",temp[k]);
        result[count++] = temp[k];
}
  }
  else if(i==7){
//ADD(":");
result[count++] = ':';
//printf("\n:");
for(j=2;j>=0;j--){
      if(temp[j]!='0')
             break;
}
for(k=j;k>=0;k--){
//          printf("\n%c",temp[k]);
              result[count++] = temp[k];
}
  }
  }

  in = 0;
  for(w=0;w<num;w++){
if((strcmp(nodes[w].address,result))==0){
                printf("\nHERE\n");
       printf("\n num :%d\n",num);
```

```
                printf("%s\n",nodes[w].address);
            nodes[w].x = msg.loc_x;
            nodes[w].y = msg.loc_y;
            in = 1;
        }
         }
         if(in == 0){
                printf("HERE2\n");
            strcpy(nodes[num].address,result);
            nodes[num].x = msg.loc_x;
            nodes[num].y = msg.loc_y;
            num++;
         }
         }
         //simple_udp_sendto(&udp_connSC, &msg, sizeof(msg), &addr);


}

-------------------------------------------------------------------------*/
//receive info from server
static void
udp_rx_SC_callback(struct simple_udp_connection *c,
                const uip_ipaddr_t *sender_addr,
                uint16_t sender_port,
                const uip_ipaddr_t *receiver_addr,
                uint16_t receiver_port,
                const uint8_t *data,
                uint16_t datalen)
{
      rpl_loc_msg_t msg = *(rpl_loc_msg_t *)data;

     LOG_INFO("Received Location Info X,Y => %d, %d\nFrom ", msg.x, msg.y);
     LOG_INFO_6ADDR(&msg.addr);
     LOG_INFO_("\n");
     seat = &msg.addr;

       x = msg.x;
       y = msg.y;
       printf("\nWeb2\n");

    int i;


       for(i=0;i<30;i++){
     result[i] = '\n';
     temp[i] = '\n';
             count=0;
       }

       uiplib_ipaddr_snprint(temp, sizeof(temp), &msg.addr);
       printf("temp is %s\n",temp);



       in = 0;
       for(w=0;w<num;w++){
     if((strcmp(nodes[w].address,temp))==0){
```

```
                printf("\nHERE\n");
            printf("\n num :%d\n",num);
                printf("%s\n",nodes[w].address);
            nodes[w].x = msg.x;
            nodes[w].y = msg.y;
            in = 1;
     }
       }
       if(in == 0){
                printf("HERE2\n");
            strcpy(nodes[num].address,temp);
            nodes[num].x = msg.x;
            nodes[num].y = msg.y;
            num++;
       }
       }



/*---------------------------------------------------------------------------*/
//add and send data to web
static
PT_THREAD(generate_routes(struct httpd_state *s))
{
   //int number = 0;
   int i;
   PSOCK_BEGIN(&s->sout);


   printf("send json to requester\n");
   ADD("{\"nodes\":[");
   int z;
   for(i=0;i<num;i++){
         if(i!=0)
     ADD(",");
         ADD("{\"names\":\"");
         for(z=0;z<strlen(nodes[i].address);z++)
               change(nodes[i].address[z]);
         ADD("\",\"X\":%d,\"Y\":%d}",nodes[i].x,nodes[i].y);

   }
   ADD("]}");
   SEND(&s->sout);
   PSOCK_END(&s->sout);
}

/*---------------------------------------------------------------------------*/
PROCESS(webserver_nogui_process, "Web server");
PROCESS_THREAD(webserver_nogui_process, ev, data)
{
   PROCESS_BEGIN();

   httpd_init();

   while(1) {
      PROCESS_WAIT_EVENT_UNTIL(ev == tcpip_event);
      httpd_appcall(data);
   }
```

```
  PROCESS_END();
}
/*---------------------------------------------------------------------------*/
httpd_simple_script_t
httpd_simple_get_script(const char *name)
{
  return generate_routes;
}
/*---------------------------------------------------------------------------*/
PROCESS_THREAD(web_sense_process, ev, data)
{
  static struct etimer periodic_timer;
  uip_ipaddr_t addr;
  static rpl_loc_msg_t msg;

  PROCESS_BEGIN();



  etimer_set(&periodic_timer, SEND_INTERVAL);

  simple_udp_register(&udp_connSC, UDP_SERVER_PORT, NULL,
                      UDP_SERVER_PORT, udp_rx_SC_callback);
  while(1) {
    PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&periodic_timer));
    etimer_reset(&periodic_timer);

    uip_create_linklocal_allnodes_mcast(&addr);
     simple_udp_sendto(&udp_connSC, &msg, sizeof(msg), &addr);
  }

  PROCESS_END();
}
```

Httpd-simple.h

```
#ifndef HTTPD_SIMPLE_H_
#define HTTPD_SIMPLE_H_

#include "contiki-net.h"

/* The current internal border router webserver ignores the requested file name */
/* and needs no per-connection output buffer, so save some RAM */
#ifndef WEBSERVER_CONF_CFS_PATHLEN
#define HTTPD_PATHLEN 2
#else /* WEBSERVER_CONF_CFS_CONNS */
#define HTTPD_PATHLEN WEBSERVER_CONF_CFS_PATHLEN
#endif /* WEBSERVER_CONF_CFS_CONNS */
```

```
struct httpd_state;
typedef char (*httpd_simple_script_t)(struct httpd_state *s);

struct httpd_state {
   struct timer timer;
   struct psock sin, sout;
   struct pt outputpt;
   char inputbuf[HTTPD_PATHLEN + 24];
/*char outputbuf[UIP_TCP_MSS]; */
   char filename[HTTPD_PATHLEN];
   httpd_simple_script_t script;
   char state;
};

void httpd_init(void);
void httpd_appcall(void *state);

httpd_simple_script_t httpd_simple_get_script(const char *name);

#define SEND_STRING(s, str) PSOCK_SEND(s, (uint8_t *)str, strlen(str))

#endif /* HTTPD_SIMPLE_H_ */
```

http-simple.c

```
#include "contiki.h"
#include "contiki-net.h"

#include <stdio.h>
#include <string.h>

#include "httpd-simple.h"
#define webserver_log_file(...)
#define webserver_log(...)

#ifndef WEBSERVER_CONF_CFS_CONNS
#define CONNS UIP_TCP_CONNS
#else /* WEBSERVER_CONF_CFS_CONNS */
#define CONNS WEBSERVER_CONF_CFS_CONNS
#endif /* WEBSERVER_CONF_CFS_CONNS */

#ifndef WEBSERVER_CONF_CFS_URLCONV
#define URLCONV 0
#else /* WEBSERVER_CONF_CFS_URLCONV */
#define URLCONV WEBSERVER_CONF_CFS_URLCONV
#endif /* WEBSERVER_CONF_CFS_URLCONV */

#define STATE_WAITING 0
#define STATE_OUTPUT  1
```

```
MEMB(conns, struct httpd_state, CONNS);

#define ISO_nl          0x0a
#define ISO_space       0x20
#define ISO_period      0x2e
#define ISO_slash       0x2f

/*---------------------------------------------------------------------------*/
static const char *NOT_FOUND = "<html><body bgcolor=\"white\">"
"<center>"
"<h1>404 - file not found</h1>"
"</center>"
"</body>"
"</html>";
/*---------------------------------------------------------------------------*/
static
PT_THREAD(send_string(struct httpd_state *s, const char *str))
{
  PSOCK_BEGIN(&s->sout);

  SEND_STRING(&s->sout, str);

  PSOCK_END(&s->sout);
}
/*---------------------------------------------------------------------------*/
const char http_content_type_json[] = "Content-type: application/json\r\n";
const char http_content_type_html[] = "Content-type: application/json\r\n\r\n";
static
PT_THREAD(send_headers(struct httpd_state *s, const char *statushdr))
{
  /* char *ptr; */

  PSOCK_BEGIN(&s->sout);

  SEND_STRING(&s->sout, statushdr);

  /* ptr = strrchr(s->filename, ISO_period); */
  /* if(ptr == NULL) { */
  /*     s->ptr = http_content_type_plain; */
  /* } else if(strcmp(http_html, ptr) == 0) { */
  /*     s->ptr = http_content_type_html; */
  /* } else if(strcmp(http_css, ptr) == 0) { */
  /*     s->ptr = http_content_type_css; */
  /* } else if(strcmp(http_png, ptr) == 0) { */
  /*     s->ptr = http_content_type_png; */
  /* } else if(strcmp(http_gif, ptr) == 0) { */
  /*     s->ptr = http_content_type_gif; */
  /* } else if(strcmp(http_jpg, ptr) == 0) { */
  /*     s->ptr = http_content_type_jpg; */
  /* } else { */
  /*     s->ptr = http_content_type_binary; */
  /* } */
  /* SEND_STRING(&s->sout, s->ptr); */
  SEND_STRING(&s->sout, http_content_type_json);
  SEND_STRING(&s->sout, http_content_type_html);
  PSOCK_END(&s->sout);
}
/*---------------------------------------------------------------------------*/
```

```
const char http_header_200[] = "HTTP/1.0 200 OK\r\nServer: Contiki/2.4
http://www.sics.se/contiki/\r\nConnection: close\r\n";
const char http_header_404[] = "HTTP/1.0 404 Not found\r\nServer: Contiki/2.4
http://www.sics.se/contiki/\r\nConnection: close\r\n";
static
PT_THREAD(handle_output(struct httpd_state *s))
{
  PT_BEGIN(&s->outputpt);

  s->script = NULL;
  s->script = httpd_simple_get_script(&s->filename[1]);
  if(s->script == NULL) {
    strncpy(s->filename, "/notfound.html", sizeof(s->filename) - 1);
    s->filename[sizeof(s->filename) - 1] = '\0';
    PT_WAIT_THREAD(&s->outputpt,
                   send_headers(s, http_header_404));
    PT_WAIT_THREAD(&s->outputpt,
                   send_string(s, NOT_FOUND));
    uip_close();
    webserver_log_file(&uip_conn->ripaddr, "404 - not found");
    PT_EXIT(&s->outputpt);
  } else {
    PT_WAIT_THREAD(&s->outputpt,
                   send_headers(s, http_header_200));
    PT_WAIT_THREAD(&s->outputpt, s->script(s));
  }
  s->script = NULL;
  PSOCK_CLOSE(&s->sout);
  PT_END(&s->outputpt);
}
/*---------------------------------------------------------------------------*/
const char http_get[] = "GET ";
const char http_index_html[] = "/index.html";

static
PT_THREAD(handle_input(struct httpd_state *s))
{
  PSOCK_BEGIN(&s->sin);

  PSOCK_READTO(&s->sin, ISO_space);

  if(strncmp(s->inputbuf, http_get, 4) != 0) {
    PSOCK_CLOSE_EXIT(&s->sin);
  }
  PSOCK_READTO(&s->sin, ISO_space);

  if(s->inputbuf[0] != ISO_slash) {
    PSOCK_CLOSE_EXIT(&s->sin);
  }

#if URLCONV
  s->inputbuf[PSOCK_DATALEN(&s->sin) - 1] = 0;
  urlconv_tofilename(s->filename, s->inputbuf, sizeof(s->filename));
#else /* URLCONV */
  if(s->inputbuf[1] == ISO_space) {
    strncpy(s->filename, http_index_html, sizeof(s->filename) - 1);
    s->filename[sizeof(s->filename) - 1] = '\0';
  } else {
```

```
      s->inputbuf[PSOCK_DATALEN(&s->sin) - 1] = 0;
      strncpy(s->filename, s->inputbuf, sizeof(s->filename));
    }
#endif /* URLCONV */

    webserver_log_file(&uip_conn->ripaddr, s->filename);

    s->state = STATE_OUTPUT;

    while(1) {
      PSOCK_READTO(&s->sin, ISO_nl);
#if 0
      if(strncmp(s->inputbuf, http_referer, 8) == 0) {
        s->inputbuf[PSOCK_DATALEN(&s->sin) - 2] = 0;
        webserver_log(s->inputbuf);
      }
#endif
    }

    PSOCK_END(&s->sin);
}
/*---------------------------------------------------------------------------*/
static void
handle_connection(struct httpd_state *s)
{
    handle_input(s);
    if(s->state == STATE_OUTPUT) {
      handle_output(s);
    }
}
/*---------------------------------------------------------------------------*/
void
httpd_appcall(void *state)
{
    struct httpd_state *s = (struct httpd_state *)state;

    if(uip_closed() || uip_aborted() || uip_timedout()) {
      if(s != NULL) {
        s->script = NULL;
        memb_free(&conns, s);
      }
    } else if(uip_connected()) {
      s = (struct httpd_state *)memb_alloc(&conns);
      if(s == NULL) {
        uip_abort();
        webserver_log_file(&uip_conn->ripaddr, "reset (no memory block)");
        return;
      }
      tcp_markconn(uip_conn, s);
      PSOCK_INIT(&s->sin, (uint8_t *)s->inputbuf, sizeof(s->inputbuf) - 1);
      PSOCK_INIT(&s->sout, (uint8_t *)s->inputbuf, sizeof(s->inputbuf) - 1);
      PT_INIT(&s->outputpt);
      s->script = NULL;
      s->state = STATE_WAITING;
      timer_set(&s->timer, CLOCK_SECOND * 10);
      handle_connection(s);
    } else if(s != NULL) {
      if(uip_poll()) {
```

```
        if(timer_expired(&s->timer)) {
          uip_abort();
          s->script = NULL;
          memb_free(&conns, s);
          webserver_log_file(&uip_conn->ripaddr, "reset (timeout)");
        }
      } else {
        timer_restart(&s->timer);
      }
      handle_connection(s);
    } else {
      uip_abort();
    }
}
/*---------------------------------------------------------------------------*/
void
httpd_init(void)
{

  tcp_listen(UIP_HTONS(80));
  memb_init(&conns);
#if URLCONV
  urlconv_init();
#endif /* URLCONV */
}
/*---------------------------------------------------------------------------*/
```

Contiki 內 Index.js

```
var express = require('express');
var request = require('request');
var app = express();
var http = require('http').Server(app);
var io = require('socket.io')(http);
var MongoClient = require('mongodb').MongoClient;
var url =
"mongodb+srv://tyson147258:55688@testuse.4dc1h.mongodb.net/tutorials?retryWrites=true&w=majority";
app.use(express.static('public'));
async function change(obj,i){
      var a = "node";
      var num = await obj.nodes[i].names.toString();
      //var mix = await a+num;
      var X = await obj.nodes[i].X;
      var Y = await obj.nodes[i].Y;
      var str = "[]";
      MongoClient.connect(url, {
      useNewUrlParser: true,
      useUnifiedTopology: true

},async function(err, db) {
      if (err) throw err;
      console.log("connect\n");
      var dbo = db.db("tutorials");

      //console.log(mix);
      var whereStr = await {"names":num};
      var newnew = await { "names" : num , "X" :   X, "Y" : Y };
      var updateStr = await {$set: { "X" :   X, "Y" : Y}};
      dbo.collection("tutorials").find(whereStr).toArray((err, res) => {
```

```
          if (err) {
             //console.log("jaijdiodmimlamiamsilsamo");
             throw err;
          }
          console.log(res);
          var gogo = JSON.stringify(res);

          console.log(gogo === str);
          if(gogo === str){
        dbo.collection("tutorials").insertOne(newnew, function(err, res){
          if (err) throw err;
          console.log(i);
          console.log("1 document insert");

          db.close();

       });
          }
          else{
               dbo.collection("tutorials").updateOne(whereStr, updateStr, function(err, res){
          if (err) throw err;
          console.log(i);
          console.log("1 document updated");

          db.close();

       });
          }

          // find() 為非同步操作，為了使查詢完後在關閉資料，所以放至此位置
          //db.close();
       });
     // dbo.collection("tutorials").updateOne(whereStr, updateStr, function(err, res) {
     // if (err) throw err;
     // console.log(num);
     // console.log("1 document updated");

   //    db.close();

   //    });
});

}
app.get('/', function(req, res){
  res.sendFile(__dirname + '/index.html');
});
io.on('connection', function(socket) {
  var dataPusher = setInterval(function () {
  request.get('http://[fd00::205:5:5:5]/',function
(err,res,body){
  if(err){
  console.log(err);
  return;
  }
  console.log(body);
  var obj = JSON.parse(body);
  console.log(obj);
  console.log(obj.nodes[0].X);
```

```
  var count = Object.keys(obj.nodes).length;
console.log(count);

  for(var i=0;i<count;i++){


      change(obj,i);

}

  });


  }, 3000);
  socket.on('disconnect', function() {
  console.log('closing');
  });
});
http.listen(3000, function(){
  console.log('listening on *:3000');
});
```

後端

Db.config.js

```
module.exports = {
        url :
"mongodb+srv://tyson147258:55688@testuse.4dc1h.mongodb.net/tutorials?retryWrites=true&w=majority"

};
```

Tutorial.controller.js

```
const db = require("../models");
const Tutorial = db.tutorials;

// Create and Save a new Tutorial
exports.create = (req, res) => {
        // Validate request
    if (!req.body.title) {
        res.status(400).send({ message: "Content can not be empty!" });
        return;
    }

    // Create a Tutorial
    const tutorial = new Tutorial({
        title: req.body.title,
        description: req.body.description,
        published: req.body.published ? req.body.published : false
```

```
    });

    // Save Tutorial in the database
    tutorial
        .save(tutorial)
        .then(data => {
            res.send(data);
        })
        .catch(err => {
            res.status(500).send({
                message:
                    err.message || "Some error occurred while creating the Tutorial."
            });
        });
};

// Retrieve all Tutorials from the database.
exports.findAll = (req, res) => {
        const title = req.query.name;
        var condition = title ? { title: { $regex: new RegExp(title)} } : {}
        Tutorial.find(condition)
            .then(data => {
                res.send(data);
            })
            .catch(err => {
                res.status(500).send({
                    message:
                        err.message || "Some error occurred while retrieving tutorials."
                });
            });
};

// Find a single Tutorial with an id
exports.findOne = (req, res) => {
        const id = req.params.id;

        Tutorial.find({ names: id })
            .then(data => {
                if (!data)
                    res.status(404).send({ message: "Not found Tutorial with id " + id });
                else res.send(data);
            })
            .catch(err => {
                res
                    .status(500)
                    .send({ message: "Error retrieving Tutorial with id=" + id });
            });
};

// Update a Tutorial by the id in the request
exports.update = (req, res) => {
        if (!req.body) {
                return res.status(400).send({
                    message: "Data to update can not be empty!"
                });
            }

            const id = req.params.id;
```

```
        Tutorial.findByIdAndUpdate(id, req.body, { useFindAndModify: false })
          .then(data => {
            if (!data) {
              res.status(404).send({
                message: `Cannot update Tutorial with id=${id}. Maybe Tutorial was not found!`
              });
            } else res.send({ message: "Tutorial was updated successfully." });
          })
          .catch(err => {
            res.status(500).send({
              message: "Error updating Tutorial with id=" + id
            });
          });
};

// Delete a Tutorial with the specified id in the request
exports.delete = (req, res) => {
    const id = req.params.id;

    Tutorial.findByIdAndRemove(id)
      .then(data => {
        if (!data) {
          res.status(404).send({
            message: `Cannot delete Tutorial with id=${id}. Maybe Tutorial was not found!`
          });
        } else {
          res.send({
            message: "Tutorial was deleted successfully!"
          });
        }
      })
      .catch(err => {
        res.status(500).send({
          message: "Could not delete Tutorial with id=" + id
        });
      });
};

// Delete all Tutorials from the database.
exports.deleteAll = (req, res) => {
    Tutorial.deleteMany({})
    .then(data => {
      res.send({
        message: `${data.deletedCount} Tutorials were deleted successfully!`
      });
    })
    .catch(err => {
      res.status(500).send({
        message:
          err.message || "Some error occurred while removing all tutorials."
      });
    });
};

// Find all published Tutorials
exports.findAllPublished = (req, res) => {
    Tutorial.find({ published: true })
```

```
        .then(data => {
           res.send(data);
        })
        .catch(err => {
           res.status(500).send({
              message:
                 err.message || "Some error occurred while retrieving tutorials."
           });
        });
};
```

Index.js

```
const dbConfig = require("../config/db.config.js");

const mongoose = require("mongoose");
mongoose.Promise = global.Promise;

const db = {};
db.mongoose = mongoose;
db.url = dbConfig.url;
db.tutorials = require("./tutorial.model.js")(mongoose);

module.exports = db;
```

tutorial.model.js

```
module.exports = mongoose => {
   const Tutorial = mongoose.model(
      "tutorial",
      mongoose.Schema(
         {
            names: String,
            X: String,
            Y: String
         },
         { timestamps: true }

      )
   );

   return Tutorial;

};
```

Turorial.routes.js

```
module.exports = app => {
        const tutorials = require("../controllers/tutorial.controller.js");
        var express = require('express');
        var router = require("express").Router();

        // Create a new Tutorial
        router.post("/", tutorials.create);

        // Retrieve all Tutorials
        router.get("/", tutorials.findAll);

        // Retrieve all published Tutorials
        router.get("/published", tutorials.findAllPublished);

        // Retrieve a single Tutorial with id
        router.get("/:id", tutorials.findOne);

        // Update a Tutorial with id
        router.put("/:id", tutorials.update);

        // Delete a Tutorial with id
        router.delete("/:id", tutorials.delete);

        // Create a new Tutorial
        router.delete("/", tutorials.deleteAll);

        app.use('/api/tutorials', router);
    };
```

Server.js

```
const express = require("express");
const bodyParser = require("body-parser");
const cors = require("cors");

const app = express();

var corsOptions = {
   origin: "http://localhost:8081"
};

app.use(cors(corsOptions));

// parse requests of content-type - application/json
app.use(bodyParser.json());

// parse requests of content-type - application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: true }));

// simple route
app.get("/", (req, res) => {
   res.json({ message: "Welcome to bezkoder application." });
```

```
});
const db = require("./app/models");
db.mongoose
   .connect(db.url, {
      useNewUrlParser: true,
      useUnifiedTopology: true
   });

// set port, listen for requests
const PORT = process.env.PORT || 8080;
require("./app/routes/turorial.routes")(app);
app.listen(PORT, () => {
   console.log(`Server is running on port ${PORT}.`);
   console.log(` http://localhost:8080/`)
});
```

前端

Index.html

```
<!DOCTYPE html>
<html lang="">
   <head>
      <meta charset="utf-8">

      <meta http-equiv="refresh" content="5">

      <meta http-equiv="X-UA-Compatible" content="IE=edge">
      <meta name="viewport" content="width=device-width,initial-scale=1.0">
      <link rel="icon" href="<%= BASE_URL %>favicon.ico">
      <title>室內定位點</title>
      <link type="text/css" rel="stylesheet" href="//unpkg.com/bootstrap/dist/css/bootstrap.min.css" />
   </head>
   <body>
   </head>
   <body>
      <noscript>
         <strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't work properly without JavaScript enabled. Please enable it to continue.</strong>
      </noscript>
      <div id="app"></div>


      <!-- built files will be auto injected -->
   </body>
</html>
```

LineChart.vue

```
<script>
import { Scatter} from 'vue-chartjs';

import TutorialDataService from "../services/TutorialDataService";
export default {
  extends: Scatter,

  data() {
    return {
      tutorials: [],
      currentTutorial: null,
      currentIndex: -1,
      title: ""

    };
  },
  methods: {
    printdata() {
      //console.log(this.tutorials.length);
      for (let i = 0; i < this.tutorials.length; i++){
        //console.log(this.tutorials[i].names);
      }
    },
    try_data(){
      let text = '[';
      for (let i = 0; i < this.tutorials.length; i++){
        if(i !== 0)
          text+=',';
        text+='{"x":'+Number(this.tutorials[i].X)+','+'"y":'+Number(this.tutorials[i].Y)+'}';
        //console.log(this.tutorials[i].names);
      }
      text+=']';
      //console.log(text);
      text = JSON.parse(text);
      return text;
    },
    try_name(){
      var text2 = [];
      for (let i = 0; i < this.tutorials.length; i++){
        text2[i] = this.tutorials[i].names;
        //console.log(this.tutorials[i].names);
      }
      return text2;
    },
    retrieveTutorials() {
      TutorialDataService.getAll()
        .then(response => {
          this.tutorials = response.data;
          console.log(response.data);
          this.printdata();
          let chartdata = {
            datasets: [
              {
              label: '室內點座標圖',
              labels: this.try_name(),
              data: this.try_data(),
```

```
                backgroundColor: 'rgb(65, 105, 225)',

                }
                ]
            }
            //console.log(chartdata.datasets[0].labels);
            this.renderChart(chartdata,{
               responsive: true,
               maintainAspectRatio: false,
               tooltips: {
                  mode: 'index',
                  callbacks: {

                        title: (tooltipItems, data) => {
                           var sum = '';
                           //var name = data.datasets[0].labels;
                           //console.log(X);
                           console.log(tooltipItems[0].index);
                           tooltipItems.forEach(function(tooltipItem) {
                                 sum += data.datasets[0].labels[tooltipItem.index];
                           });
                           return 'IPv6 address :' + sum;
                        },
                  },
                  footerFontStyle: 'normal'
               }




            });
         })
         .catch(e => {
            console.log(e);
         });
      },


   },
   mounted() {
      // Overwriting base render method with actual data.

      this.retrieveTutorials();




   }
}
</script>
```

TutorialList.vue

```
<template>
  <div class="list row">
    <div class="col-md-8">
      <div class="input-group mb-3">
        <input type="text" class="form-control" placeholder="Search by IPv6 address"
          v-model="title"/>
        <div class="input-group-append">
          <button class="btn btn-outline-secondary" type="button"
            @click="searchTitle"
          >
            Search
          </button>
        </div>
      </div>
    </div>
    <div class="col-md-6">
      <h4>Nodes List</h4>
      <ul class="list-group">
        <li class="list-group-item"
          :class="{ active: index == currentIndex }"
          v-for="(tutorial, index) in tutorials"
          :key="index"
          @click="setActiveTutorial(tutorial, index)"
        >
          {{ tutorial.names }}
        </li>
      </ul>
    </div>
    <div class="col-md-6">
      <div v-if="currentTutorial">
        <h4>Node</h4>
        <div>
          <label><strong>IPv6 address:</strong></label> {{ currentTutorial.names }}
        </div>
        <div>
          <label><strong>X 座標:</strong></label> {{ currentTutorial.X }}
        </div>
        <div>
          <label><strong>Y 座標:</strong></label> {{ currentTutorial.Y}}
        </div>


      </div>
      <div v-else>
        <br />
        <p>Please click on Node List or Search...</p>
      </div>
    </div>
  </div>
</template>

<script>
import TutorialDataService from "../services/TutorialDataService";

export default {
  name: "tutorials-list",
```

```
  data() {
    return {
      tutorials: [],
      currentTutorial: null,
      currentIndex: -1,
      title: ""
    };
  },
  methods: {
    retrieveTutorials() {
      TutorialDataService.getAll()
        .then(response => {
          this.tutorials = response.data;
          console.log(response.data);
        })
        .catch(e => {
          console.log(e);
        });
    },

    refreshList() {
      this.retrieveTutorials();
      this.currentTutorial = null;
      this.currentIndex = -1;
    },

    setActiveTutorial(tutorial, index) {
      this.currentTutorial = tutorial;
      this.currentIndex = index;
    },

    removeAllTutorials() {
      TutorialDataService.deleteAll()
        .then(response => {
          console.log(response.data);
          this.refreshList();
        })
        .catch(e => {
          console.log(e);
        });
    },

    searchTitle() {
      TutorialDataService.findByTitle(this.title)
        .then(response => {
          this.tutorials = response.data;
          console.log(response.data);
        })
        .catch(e => {
          console.log(e);
        });
    }
  },
  mounted() {
    this.retrieveTutorials();
  }
};
</script>
```

```
<style>
.list {
  text-align: left;
  max-width: 750px;
  margin: auto;
}
</style>
```

TutorialDataService.vue

```
import http from "../http-common";

class TutorialDataService {
  getAll() {
    return http.get("/tutorials");
  }

  get(id) {
    return http.get(`/tutorials/${id}`);
  }

  create(data) {
    return http.post("/tutorials", data);
  }

  update(id, data) {
    return http.put(`/tutorials/${id}`, data);
  }

  delete(id) {
    return http.delete(`/tutorials/${id}`);
  }

  deleteAll() {
    return http.delete(`/tutorials`);
  }

  findByTitle(title) {
    return http.get(`/tutorials/${title}`);
  }
}

export default new TutorialDataService();
```

App.vue

```
<template>
  <div id="app">
```

```
      <nav class="navbar navbar-expand navbar-dark bg-dark">
        <router-link to="/" class="navbar-brand">室内定位</router-link>

      </nav>

      <div class="container mt-3">
        <router-view />
      </div>
      <div class="row justify-content-center align-items-stretch" >
            <div class="col-md-8 mt-2 " style="width:1750px; height:400px">
              <LineChart/>
            </div>
      </div>
    </div>
</template>

<script>

import LineChart from './components/LineChart'

export default {
  name: "app",
  components: {
    LineChart,
  }
}
</script>

<style>
#app {
  font-family: Avenir, Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
  margin-top: 0px;
}
</style>
```

http-common.js

```
import axios from "axios";

export default axios.create({
  baseURL: "http://localhost:8080/api",
  headers: {
    "Content-type": "application/json"
  }
});
```

Main.js

```
import Vue from 'vue'
import App from './App.vue'
import router from './router'
import { BootstrapVue, IconsPlugin } from 'bootstrap-vue'
// Install BootstrapVue
Vue.use(BootstrapVue)
// Optionally install the BootstrapVue icon components plugin
Vue.use(IconsPlugin)
import 'bootstrap/dist/css/bootstrap.css'
import 'bootstrap-vue/dist/bootstrap-vue.css'
Vue.config.productionTip = false

new Vue({
  router,
  render: h => h(App),
}).$mount('#app')
```

Router.js

```
import Vue from "vue";
import Router from "vue-router";

Vue.use(Router);

export default new Router({
  mode: "history",
  routes: [
    {
      path: "/",
      alias: "/tutorials",
      name: "tutorials",
      component: () => import("./components/TutorialsList")
    },
    {
      path: "/tutorials/:id",
      name: "tutorial-details",
      component: () => import("./components/Tutorial")
    },
    {
      path: "/add",
      name: "add",
      component: () => import("./components/AddTutorial")
    }
  ]
});
```