APPENDIX SOURCE CODE

MICRO-CONTROLLER : SOURCE CODE

```
--------------------------------------------------------------------------------------------

#include <SoftwareSerial.h>
#define switch1 2     //lights
#define switch2 3     //fan
#define switch3 4     //night bulb
#define switch4 5     //socket
#define temp_pin A0
#define brightness_pin A1
#define motion_pin A2
#define buzzer 9
#define sleep_buzzer 8

boolean ss1 = false;
boolean ss2 = false;
boolean ss3 = false;
boolean ss4 = false;

boolean auto_mode = true;
boolean secure_mode = false;
boolean sleep_mode = false;

int seconds = 0;
int minutes = 0;
int hours = 0;
long current_time = 0;

int minutes1 = -1;
boolean action1 = false;
int minutes2 = -1;
boolean action2 = false;
int minutes3 = -1;
boolean action3 = false;
int minutes4 = -1;
boolean action4 = false;


int auto_cutoff_time = 2;                 //default power cutoff time 2 mins
int current_cutoff_time = -1;
int room_brightness = 149;
int threshold_brightness = 150;
int room_temprature = 21;
int threshold_temprature = 20;
int temprature_compensation = 5;
boolean motion = false;
boolean alarm = false;
int alarm_time = -1;
int default_alarm_time = 10;              //default Alarm time 10 mins
int sleep_time = -1;                      //minutes
int default_sleep_alarm_time = 300;       // in seconds
int sleep_alarm_time = -1;                // in seconds
boolean sleep_alarm = false;


/*protocols for receiving command
toggle        :    #switch-no    :  #1, #2, #3, #4
command       :    !command     :  !data,!secure,!auto,!tfa,!toa, !sleep
```

```
threshold_brightness, threshold_temprature, auto_cutoff_time, default_alarm_time
timer          :    @switch_no:action:minutes   :   @1:0:384, @2:1:33, @s:0:450,
@s:0:-1, @2:1:-1 etc.
*/

SoftwareSerial BTSerial(10, 11); // RX | TX
String instream;

void setup()
{ Serial.begin(9600);
  Serial.println("Enter AutoMate");
  BTSerial.begin(38400);
  //analogReference(DEFAULT);    //INTERNAL FOR 1.1 V REFERENCE  //DEFAULT 5V REFERENCE
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(A3, OUTPUT);
  pinMode(A4, OUTPUT);
  digitalWrite(6, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(A3, HIGH);
  digitalWrite(A4, LOW);}

void loop()
{  //time
  if (millis() - current_time >= 1000)
  { current_time = millis();
    if (seconds < 59)
      seconds++;
    else if (seconds == 59)
    { seconds = 0;
      if (minutes < 59)
        minutes++;
      else if (minutes == 59)
      { minutes = 0;
        if (hours < 23)
          hours++;
        else if (hours == 23)
          hours = 0; }

      if (minutes1 >= 0)
        minutes1--;
      if (minutes2 >= 0)
        minutes2--;
      if (minutes3 >= 0)
        minutes3--;
      if (minutes4 >= 0)
        minutes4--;
      if (alarm_time >= 0)
        alarm_time--;
      if (current_cutoff_time >= 0)
        current_cutoff_time--;
      if (sleep_time >= 0)
        sleep_time--;
    }
    if (sleep_alarm_time >= 0)
      sleep_alarm_time--;
  }

  //  Keep reading from HC-05
  if (BTSerial.available())
  {
    instream = "";
    while (BTSerial.available())
      instream += char(BTSerial.read());
    Serial.println(instream);
    BTSerial.flush();
    if (instream[0] == '#')
    { if (instream == "#1")
        toggle(switch1);
      else if (instream == "#2")
        toggle(switch2);
      else if (instream == "#3")
        toggle(switch3);
      else if (instream == "#4")
        toggle(switch4);
    }

    else if (instream[0] == '!')
    { if (instream == "!tfa")
      turnalloff();
      else if (instream == "!toa")
      turnallon();
      else if (instream == "!auto")
      { if (auto_mode == true)
          auto_mode = false;
        else if (auto_mode == false)
          auto_mode = true;}
      else if (instream == "!secure")
      { if (secure_mode == true)
        { secure_mode = false;
          alarm = false; }
        else if (secure_mode == false)
          secure_mode = true; }
      else if (instream == "!sleep")
      { if (sleep_mode == true)
        sleep_mode = false;
        else if (secure_mode == false)
        { sleep_mode = true;
          auto_mode = true; } }
      else if (instream == "!data")
      send_data();
  }

    else if (instream[0] == '@')
    {if (instream[1] == '1')
      { if (instream[3] == '0')
          action1 = false;
        else if (instream[3] == '1')
          action1 = true;
        minutes1 = instream.substring(5).toInt();}
      else if (instream[1] == '2')
      { if (instream[3] == '0')
          action2 = false;
        else if (instream[3] == '1')
          action2 = true;
        minutes2 = instream.substring(5).toInt();}
      else if (instream[1] == '3')
      { if (instream[3] == '0')
          action3 = false;
        else if (instream[3] == '1')
          action3 = true;
        minutes3 = instream.substring(5).toInt();}
      else if (instream[1] == '4')
      { if (instream[3] == '0')
          action4 = false;
        else if (instream[3] == '1')
          action4 = true;
        minutes4 = instream.substring(5).toInt();}
      else if (instream[1] == 's')
      {
        sleep_time = instream.substring(5).toInt();;
        if(sleep_time > 0)
        sleep_mode = true;
        else if (sleep_time=-1)
        {sleep_mode = false;
        sleep_alarm_time = 0;}
       }

    }

  }
  motion = digitalRead(motion_pin);
  room_brightness = constrain( map(analogRead(brightness_pin), 100, 900, 0, 255), 0,
  255);
  // room_temprature = (analogRead(temp_pin)) * 4.8828 / 10;          //  T degree
  celcious = (value*step_size)/10     step_size=Aref*1000/1024 mV
  10mV/degree_celcious
  timer_action();
  secure_modefn();
  automaticfn();
  delay(10);
}


void toggle(int sw)
{if (sw == switch1)
  {if (ss1 == false)
    {digitalWrite(switch1, HIGH);
     ss1 = true;}
    else if (ss1 == true)
    { digitalWrite(switch1, LOW);
      ss1 = false;}
    delay(100);
  }

  if (sw == switch2)
  {if (ss2 == false)
    { digitalWrite(switch2, HIGH);
      ss2 = true;}
    else if (ss2 == true)
    { digitalWrite(switch2, LOW);
      ss2 = false;}
    delay(100);
    }

    if (sw == switch3)
    { if (ss3 == false)
      { digitalWrite(switch3, HIGH);
        ss3 = true;}
      else if (ss3 == true)
      { digitalWrite(switch3, LOW);
        ss3 = false;}
      delay(100);
    }

    if (sw == switch4)
    { if (ss4 == false)
      { digitalWrite(switch4, HIGH);
        ss4 = true;}
      else if (ss4 == true)
      { digitalWrite(switch4, LOW);
        ss4 = false;}
      delay(100);
    }
  }

  void turnalloff()
  { digitalWrite(switch1, LOW);
    ss1 = false;
    digitalWrite(switch2, LOW);
    ss2 = false;
    digitalWrite(switch3, LOW);
    ss3 = false;
    digitalWrite(switch4, LOW);
    ss4 = false;
  }

  void turnallon()
  { digitalWrite(switch1, HIGH);
    ss1 = true;
    digitalWrite(switch2, HIGH);
    ss2 = true;
    digitalWrite(switch3, HIGH);
    ss3 = true;
    digitalWrite(switch4, HIGH);
    ss4 = true;
  }

  void timer_action()
  {if (minutes1 == 0)
    { if (action1 == true)
      { digitalWrite(switch1, HIGH);
        ss1 = HIGH;}
      else if (action1 == false)
      { digitalWrite(switch1, LOW);
        ss1 = LOW;}
    }

    if (minutes2 == 0)
    { if (action2 == true)
      { digitalWrite(switch2, HIGH);
        ss2 = HIGH;}
      else if (action2 == false)
      { digitalWrite(switch2, LOW);
        ss2 = LOW;}
    }
    if (minutes3 == 0)
    { if (action3 == true)
      { digitalWrite(switch3, HIGH);
        ss3 = HIGH;}
      else if (action3 == false)
      { digitalWrite(switch3, LOW);
        ss3 = LOW;}
    }
    if (minutes4 == 0)
    { if (action4 == true)
      { digitalWrite(switch4, HIGH);
        ss4 = HIGH;}
      else if (action4 == false)
      { digitalWrite(switch4, LOW);
        ss4 = LOW;}
    }
  }


  void secure_modefn()
  { if (secure_mode == true)
    { if (alarm == true && alarm_time == 0)
      { digitalWrite(buzzer, LOW);
        alarm = false;}
      else if (alarm == true)
       ;
      else if (motion == true)
      { digitalWrite(buzzer, HIGH);
        alarm = true;
        alarm_time = default_alarm_time;}
    }
    if (secure_mode == false)
    { digitalWrite(buzzer, LOW);
      alarm = false;
      alarm_time = -1;}
  }

  void automaticfn()
  {  //auto_on
    if (auto_mode == true && sleep_mode == false)
    { if (room_brightness <= threshold_brightness && ss1 == false && motion == true)
      { digitalWrite(switch1, HIGH);
        ss1 = true;}
      if (room_temprature >= threshold_temprature && ss2 == false && motion == true)
      { digitalWrite(switch2, HIGH);
        ss2 = true;}
      else if (room_temprature < (threshold_temprature - temprature_compensation) && ss2
      == HIGH && motion == true)
      { digitalWrite(switch2, LOW);
        ss2 = false;}

  }
    //auto_cutoff
    if (auto_mode == true && sleep_mode == false)
    {   if (motion == true || room_brightness <= 150)
        current_cutoff_time = auto_cutoff_time;
      else if (current_cutoff_time == 0)
      { digitalWrite(switch1, LOW);
        ss1 = false;
        digitalWrite(switch2, LOW);
        ss2 = false; }
    }

    else if (sleep_mode == true)
    { if (sleep_time > 0 )
      { digitalWrite(switch1, LOW);
        ss1 = false;
        if (room_temprature >= threshold_temprature)
        { digitalWrite(switch2, HIGH);
          ss2 = true; }
        if (room_temprature < (threshold_temprature - temprature_compensation))
        { digitalWrite(switch2, LOW);
          ss2 = false;}
      }
      else if (sleep_time == 0)
      { digitalWrite(sleep_buzzer, HIGH);
        sleep_alarm = true;
        sleep_alarm_time = default_sleep_alarm_time;
        sleep_mode = false;}
    }

    if (sleep_alarm_time == 0)
    { digitalWrite(sleep_buzzer, LOW);
      sleep_alarm = false;}
  }

  void send_data()
  { /*protocols for sending data to android
    Temperature = ** degree celcious
    Brightness level = **
    Secure Mode = ON/OFF
    Auto Mode = ON/OFF
    Sleep Mode = ON/OFF

    Switch state 1 = ON/OFF
    Switch state 2 = ON/OFF
    Switch state 3 = ON/OFF
    Switch state 4 = ON/OFF

    Time to switch
    ON/OFF 1 : minutes1 mins.
    ON/OFF 2 : minutes2 mins.
    ON/OFF 3 : minutes3 mins.
    ON/OFF 4 : minutes4 mins.
    Sleep Mode :

    Threshold temperature : ** degree celcious
    Threshold brightness : ***
    Cutoff time : ** mins.
    Alarm time  : *** mins.
    Sleep Alarm time : ***mins.
    */

    BTSerial.print('Temprature : ');
    BTSerial.print(room_temprature);
    BTSerial.println(' degree celcious');
    BTSerial.print('Brightness level : ');
```

```
BTSerial.print(room_brightness);
BTSerial.println(' ');
BTSerial.print('Secure Mode : ');
if (secure_mode == true)
  BTSerial.println('ON');
else BTSerial.println('OFF');
BTSerial.print('Auto Mode : ');
if (auto_mode == true)
  BTSerial.println('ON');
else BTSerial.println('OFF');
 BTSerial.print('Sleep Mode : ');
if (sleep_mode == true)
  BTSerial.println('ON');
else BTSerial.println('OFF');
BTSerial.println(' ');

BTSerial.print('Switch state 1 : ');
if (ss1 == true)
  BTSerial.println('ON');
else BTSerial.println('OFF');
BTSerial.print('Switch state 2 : ');
if (ss2 == true)
  BTSerial.println('ON');
else BTSerial.println('OFF');
BTSerial.print('Switch state 3 : ');
if (ss3 == true)
  BTSerial.println('ON');
else BTSerial.println('OFF');
BTSerial.print('Switch state 4 : ');
if (ss4 == true)
  BTSerial.print('ON');
else BTSerial.println('OFF');
BTSerial.println(' ');

BTSerial.print('Time to switch');
if (action1 == true)
  BTSerial.print('ON');
else BTSerial.println('OFF');
BTSerial.print(' 1 : ');
BTSerial.println(minutes1);
if (action2 == true)
  BTSerial.print('ON');
else BTSerial.println('OFF');
BTSerial.print(' 2 : ');
BTSerial.println(minutes2);
if (action3 == true)
  BTSerial.print('ON');
else BTSerial.println('OFF');
BTSerial.print(' 3 : ');
BTSerial.println(minutes3);
if (action4 == true)
  BTSerial.print('ON');
else BTSerial.println('OFF');
BTSerial.print(' 4 : ');
BTSerial.println(minutes4);
BTSerial.println(' ');
BTSerial.print(' Sleep Mode : ');
BTSerial.println(sleep_time);
BTSerial.println(' ');
```

```
    BTSerial.print('Threshold Temprature : ');
    BTSerial.print(threshold_temprature);
    BTSerial.println(' degree celcious');
    BTSerial.print('Threshold Brightness : ');
    BTSerial.println(threshold_brightness);
    BTSerial.print('Cut Off time : ');
    BTSerial.print(auto_cutoff_time);
    BTSerial.println(' mins.');
    BTSerial.print('Alarm time : ');
    BTSerial.print(alarm_time);
    BTSerial.println(' mins.');
    BTSerial.print('Sleep Alarm time : ');
    BTSerial.print(default_sleep_alarm_time);
    BTSerial.println(' sec.');
    BTSerial.print('#');
  }
```