```
#include <RFID.h>
#include <MFRC522.h>

#include <SPI.h>

#define SAD 10
#define RST 5

MFRC522 nfc(SAD, RST);
/*Modified by: BehindTheSciences.com */

#include <LiquidCrystal.h>
//#define RS 12
#define RS 6
#define RW 2
//#define EN 11
#define EN 4
//#define VO 13
#define VO 5
#define D4 7
#define D5 8
//#define D6 9
#define D6 A0
//#define D7 10
#define D7 3
int i = 1;
// initialize the library with the numbers of the interface pins
//LiquidCrystal lcd(12, 11, 7, 8, 9, 10);
LiquidCrystal lcd(RS, EN, D4, D5, D6, D7);

void setup() {
  /*Initialisation of the LCD */
  pinMode(D4, OUTPUT);
  pinMode(D5, OUTPUT);
  pinMode(D6, OUTPUT);
  pinMode(D7, OUTPUT);

  pinMode(EN, OUTPUT);
```

```
digitalWrite(EN, HIGH);
pinMode(RS, OUTPUT);
digitalWrite(RS, LOW);
pinMode(VO, OUTPUT);
digitalWrite(VO, LOW);

pinMode(RW, OUTPUT);
digitalWrite(RW, LOW);

delay(20);
digitalWrite(D7, LOW);
digitalWrite(D6, LOW);
digitalWrite(D5, HIGH);
digitalWrite(D4, HIGH);
delay(10);
digitalWrite(D7, LOW);
digitalWrite(D6, LOW);
digitalWrite(D5, HIGH);
digitalWrite(D4, LOW);

digitalWrite(D7, LOW);
digitalWrite(D6, LOW);
digitalWrite(D5, HIGH);
digitalWrite(D4, LOW);

digitalWrite(D7, HIGH);
digitalWrite(D6, HIGH);

digitalWrite(D7, LOW);
digitalWrite(D6, LOW);
digitalWrite(D5, LOW);
digitalWrite(D4, LOW);

digitalWrite(D7, HIGH);
digitalWrite(D6, LOW);
digitalWrite(D5, LOW);
digitalWrite(D4, LOW);

digitalWrite(D7, LOW);
digitalWrite(D6, LOW);
```

```
 digitalWrite(D5, LOW);
 digitalWrite(D4, LOW);

 digitalWrite(D7, LOW);
 digitalWrite(D6, LOW);
 digitalWrite(D5, LOW);
 digitalWrite(D4, HIGH);

 digitalWrite(D7, LOW);
 digitalWrite(D6, LOW);
 digitalWrite(D5, LOW);
 digitalWrite(D4, LOW);

 digitalWrite(D7, LOW);
 digitalWrite(D6, HIGH);
 digitalWrite(D5, HIGH);
 digitalWrite(D4, HIGH);
/*----------------------*/
 lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.setCursor(3, 0);
  lcd.print("Behind The");
  lcd.setCursor(4, 1);
  lcd.print("Sciences");
  delay(30);
   lcd.clear();
 lcd.setCursor(1, 0);

  SPI.begin();
  // Read a fast as possible. There is a limit for how long we are
  // allowed to read from the tags.
  Serial.begin(115200);

  Serial.println("Looking for MFRC522.");
  nfc.begin();

  // Get the firmware version of the RFID chip
  byte version = nfc.getFirmwareVersion();
  if (! version) {
    Serial.print("Didn't find MFRC522 board.");
```

```
    while(1); //halt
  }

  Serial.print("Found chip MFRC522 ");
  Serial.print("Firmware ver. 0x");
  Serial.print(version, HEX);
  Serial.println(".");
}

byte keyA[6] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, };
byte keyB[6] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, };

void loop() {
  byte status;
  byte data[MAX_LEN];
  byte serial[5];
  int i, j, pos;

  // Send a general request out into the aether. If there is a
tag in
  // the area it will respond and the status will be MI_OK.
  status = nfc.requestTag(MF1_REQIDL, data);

  if (status == MI_OK) {
    Serial.println("Tag detected.");
    Serial.print("Type: ");
    Serial.print(data[0], HEX);
    Serial.print(", ");
    Serial.println(data[1], HEX);

    // calculate the anti-collision value for the currently
detected
    // tag and write the serial into the data array.
    status = nfc.antiCollision(data);
    memcpy(serial, data, 5);

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("SerialNo:");
    Serial.println("The serial nb of the tag is:");
```

```
    for (i = 0; i < 3; i++) {
      //lcd.setCursor(i, 0);
      lcd.print(serial[i],HEX);
      Serial.print(serial[i], HEX);
      Serial.print(", ");
    }
    Serial.println(serial[3], HEX);
    lcd.print(serial[3], HEX);

    // Select the tag that we want to talk to. If we don't do
this the
    // chip does not know which tag it should talk if there
should be
    // any other tags in the area..
    nfc.selectTag(serial);

    // Assuming that there are only 64 blocks of memory in this
chip.
    for (i = 0; i < 64; i++) {
      // Try to authenticate each block first with the A key.
      status = nfc.authenticate(MF1_AUTHENT1A, i, keyA, serial);
      if (status == MI_OK) {
        Serial.print("Authenticated block nb. 0x");
        Serial.print(i, HEX);
        Serial.println(" with key A.");
        // Reading block i from the tag into data.
        status = nfc.readFromTag(i, data);
        if (status == MI_OK) {
          // If there was no error when reading; print all the hex
          // values in the data.
          for (j = 0; j < 15; j++) {
            Serial.print(data[j], HEX);
            Serial.print(", ");
          }
          Serial.println(data[15], HEX);
        } else {
          Serial.println("Read failed.");
        }
      } else {
        // If we could not authenticate with the A key, we will
```

```
try
        // the B key.
        status = nfc.authenticate(MF1_AUTHENT1B, i, keyB, serial);
        if (status == MI_OK) {
          Serial.print("Authenticated block nb. 0x");
          Serial.print(i, HEX);
          Serial.println(" with key B.");
          status = nfc.readFromTag(i, data);
          if (status == MI_OK) {
            for (j = 0; j < 15; j++) {
              Serial.print(data[j], HEX);
              Serial.print(", ");
            }
            Serial.println(data[15], HEX);
          } else {
            Serial.println("Read failed.");
          }
        } else {
          Serial.print("Access denied at block nb. 0x");
          Serial.println(i, HEX);
        }
      }
    }

    // Stop the tag and get ready for reading a new tag.
    nfc.haltTag();
  }
  delay(2000);
}
```