```cpp
#include <SPI.h>
#include <MFRC522.h>
#include <LiquidCrystal.h>

// Pin definitions
#define RST_PIN 9
#define SS_PIN 10
#define BUZZER_PIN 8

// Create instances
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.
LiquidCrystal lcd(7, 6, 5, 4, 3, 2); // Create LCD instance.

// Predefined RFID tag UIDs (example)
byte knownUIDs[][4] = {
  {0x12, 0x34, 0x56, 0x78},
  {0xAB, 0xCD, 0xEF, 0x90}
};
const int knownUIDCount = sizeof(knownUIDs) / sizeof(knownUIDs[0]);

// Function to compare UIDs
bool compareUID(byte* uid, byte knownUID[]) {
  for (int i = 0; i < 4; i++) {
    if (uid[i] != knownUID[i]) {
      return false;
    }
  }
  return true;
}

void setup() {
  // Initialize serial communication
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();

```

```
  // Initialize LCD
  lcd.begin(16, 2);
  lcd.print("RFID Attendance");

  // Initialize Buzzer
  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, LOW);
}

void loop() {
  // Look for new RFID cards
  if (!mfrc522.PICC_IsNewCardPresent()) {
    return;
  }
  if (!mfrc522.PICC_ReadCardSerial()) {
    return;
  }

  // Print UID
  Serial.print("UID tag: ");
  String content = "";
  byte* uid = mfrc522.uid.uidByte;
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    Serial.print(uid[i] < 0x10 ? " 0" : " ");
    Serial.print(uid[i], HEX);
    content.concat(String(uid[i] < 0x10 ? " 0" : " "));
    content.concat(String(uid[i], HEX));
  }
  Serial.println();

  // Compare with known UIDs
  bool knownUID = false;
  for (int i = 0; i < knownUIDCount; i++) {
    if (compareUID(uid, knownUIDs[i])) {
      knownUID = true;
      break;
    }
  }

  // Display result on LCD and buzz
  lcd.clear();
  if (knownUID) {
    lcd.print("Access Granted");
    digitalWrite(BUZZER_PIN, HIGH);
    delay(200);
    digitalWrite(BUZZER_PIN, LOW);
  } else {
    lcd.print("Access Denied");
  }

  // Halt PICC
  mfrc522.PICC_HaltA();
}
```