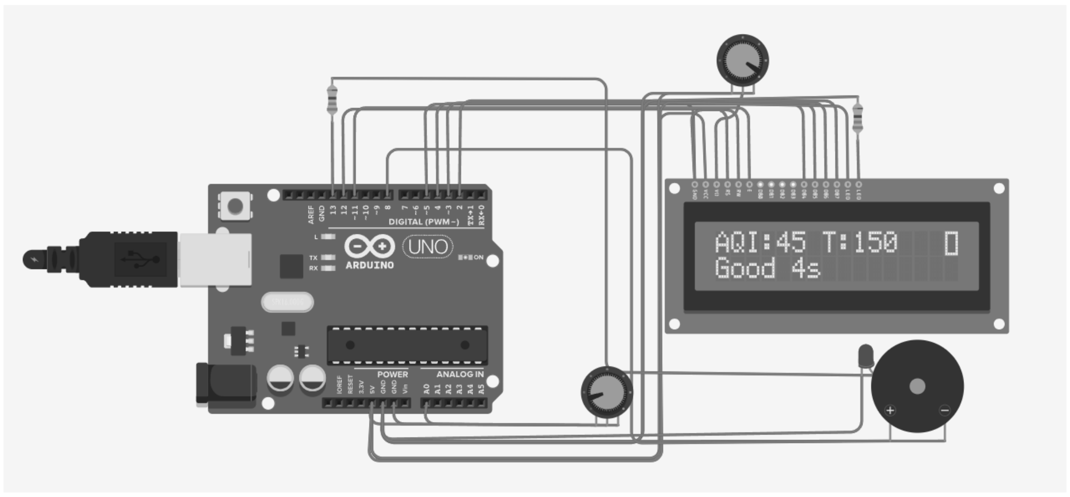
PROGRAM:

```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
const int airQualitySensorPin = A0;
const int airPurifierPin = 13;
const int buttonPin = 7;
const int buzzerPin = 8;
const int GOOD_THRESHOLD = 50;
const int MODERATE_THRESHOLD = 100;
const int UNHEALTHY_SG_THRESHOLD = 150;
const int UNHEALTHY_THRESHOLD = 200;
const int VERY_UNHEALTHY_THRESHOLD = 300;
int airQualityIndex = 0;
String airQualityStatus = "Good";
int purifierThreshold = 150;
bool purifierState = false;
bool lastPurifierState = false;
unsigned long purifierStartTime = 0;
unsigned long totalPurifierRuntime = 0;
const int ACTIVATION_SOUND_FREQ = 1000;
const int DEACTIVATION_SOUND_FREQ = 800;
const int SOUND_DURATION = 300;
unsigned long lastDisplayUpdate = 0;
const int displayUpdateInterval = 500;
unsigned long lastBreathTime = 0;
const int breathCycle = 2000;
bool buttonPressed = false;
unsigned long lastButtonPressTime = 0;
const int debounceDelay = 50;
const int longPressDuration = 2000;
byte airQualityChar[8];

void setup() {
  pinMode(airPurifierPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(buzzerPin, OUTPUT);
  lcd.begin(16, 2);
  lcd.createChar(0, airQualityChar);
  showStartupMessage();
  Serial.begin(9600);
}

void loop() {
  readSensorData();
  checkButton();
  controlPurifier();
  updateDisplay();
  delay(20);
}

void readSensorData() {
  int sensorValue = analogRead(airQualitySensorPin);
  int mappedAQI = map(sensorValue, 0, 1023, 0, 500);
  airQualityIndex = mappedAQI;
  if (airQualityIndex < GOOD_THRESHOLD)
airQualityStatus = "Good";

  else if (airQualityIndex <
MODERATE_THRESHOLD) airQualityStatus
= "Moderate";

  else if (airQualityIndex <
UNHEALTHY_SG_THRESHOLD)
airQualityStatus = "Unhealthy-SG";

  else if (airQualityIndex <
UNHEALTHY_THRESHOLD) airQualityStatus
= "Unhealthy";

  else if (airQualityIndex <
VERY_UNHEALTHY_THRESHOLD)
airQualityStatus = "Very Unhealthy";

  else airQualityStatus = "Hazardous";

}


void controlPurifier() {

  bool shouldBeOn = airQualityIndex >=
purifierThreshold;

  if (shouldBeOn && !purifierState) {

    purifierState = true;

    purifierStartTime = millis();

    playActivationSound();

    Serial.println("Purifier ON - Played
activation sound");

  }

  else if (!shouldBeOn && purifierState) {

    purifierState = false;

    totalPurifierRuntime += millis() -
purifierStartTime;

    playDeactivationSound();

    Serial.println("Purifier OFF - Played
deactivation sound");

    digitalWrite(airPurifierPin, LOW);

  }

  if (purifierState) {

    breathingLED();

  }

}


void playActivationSound() {

  tone(buzzerPin,
ACTIVATION_SOUND_FREQ,
SOUND_DURATION);

  delay(SOUND_DURATION);

  noTone(buzzerPin);

}


void playDeactivationSound() {

  tone(buzzerPin,
DEACTIVATION_SOUND_FREQ,
SOUND_DURATION);

  delay(SOUND_DURATION);

  noTone(buzzerPin);

}


void breathingLED() {

  unsigned long currentMillis = millis();

  float phase = (currentMillis -
lastBreathTime) * 2 * PI / breathCycle;

  int brightness = 128 + 127 * sin(phase);

  analogWrite(airPurifierPin, brightness);

  if (currentMillis - lastBreathTime >=
breathCycle) {

    lastBreathTime = currentMillis;

  }

}
```

```
void checkButton() {
  int buttonState = digitalRead(buttonPin);
  if (buttonState == LOW &&
!buttonPressed) {
    buttonPressed = true;
    lastButtonPressTime = millis();
  }
  else if (buttonState == HIGH &&
buttonPressed) {
    buttonPressed = false;
    unsigned long pressDuration = millis() -
lastButtonPressTime;
    if (pressDuration > debounceDelay &&
pressDuration < longPressDuration) {
      Serial.println("Button pressed - No
mode switching in this version");
    }
    else if (pressDuration >=
longPressDuration) {
      adjustThreshold();
    }
  }
}


void adjustThreshold() {
  Serial.println("Threshold adjustment
mode activated");
  lcd.clear();
  lcd.print("Adjust Threshold");
  lcd.setCursor(0, 1);
  lcd.print("Current: ");
  lcd.print(purifierThreshold);
  unsigned long startTime = millis();
  while (millis() - startTime < 5000) {
    int newThreshold =
map(analogRead(airQualitySensorPin), 0,
1023, 50, 300);
    if (abs(newThreshold -
purifierThreshold) > 5) {
      purifierThreshold = newThreshold;
      lcd.setCursor(9, 1);
      lcd.print("    ");
      lcd.setCursor(9, 1);
      lcd.print(purifierThreshold);
    }
    delay(100);
  }
  Serial.print("New threshold set: ");
  Serial.println(purifierThreshold);
}


void updateDisplay() {
  if (millis() - lastDisplayUpdate >=
displayUpdateInterval) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("AQI:");
    lcd.print(airQualityIndex);
    lcd.print(" T:");
    lcd.print(purifierThreshold);
    lcd.setCursor(0, 1);
    lcd.print(airQualityStatus);
    lcd.print(" ");
    unsigned long runtime = purifierState ?
      (totalPurifierRuntime + millis() -
purifierStartTime) / 1000 :
      totalPurifierRuntime / 1000;
    lcd.print(runtime);
    lcd.print("s");
    updateAirQualityIndicator();
    lcd.setCursor(15, 0);
    lcd.write(byte(0));
    lastDisplayUpdate = millis();
  }
}

void updateAirQualityIndicator() {
  for (int i = 0; i < 8; i++) airQualityChar[i] =
B00000;
  int bars = 1;
  if (airQualityIndex >=
MODERATE_THRESHOLD) bars = 2;
  if (airQualityIndex >=
UNHEALTHY_SG_THRESHOLD) bars = 3;
  if (airQualityIndex >=
UNHEALTHY_THRESHOLD) bars = 4;
  if (airQualityIndex >=
VERY_UNHEALTHY_THRESHOLD) bars = 5;
  for (int i = 0; i < bars; i++) {
    airQualityChar[7 - i] = B11111;
  }
  if (bars < 5) {
    airQualityChar[0] = B11111;
    airQualityChar[7] = B11111;
    for (int i = 1; i < 7; i++) {
      if (airQualityChar[i] == B00000)
airQualityChar[i] = B10001;
    }
  }
  lcd.createChar(0, airQualityChar);
}

void showStartupMessage() {
  lcd.setCursor(0, 0);
  lcd.print("Air Quality Pro");
  lcd.setCursor(0, 1);
  lcd.print("With Sound Alerts");
  tone(buzzerPin, 1500, 200);
  delay(200);
  tone(buzzerPin, 2000, 200);
  delay(500);
  noTone(buzzerPin);
  delay(1500);
  lcd.clear();
}
```

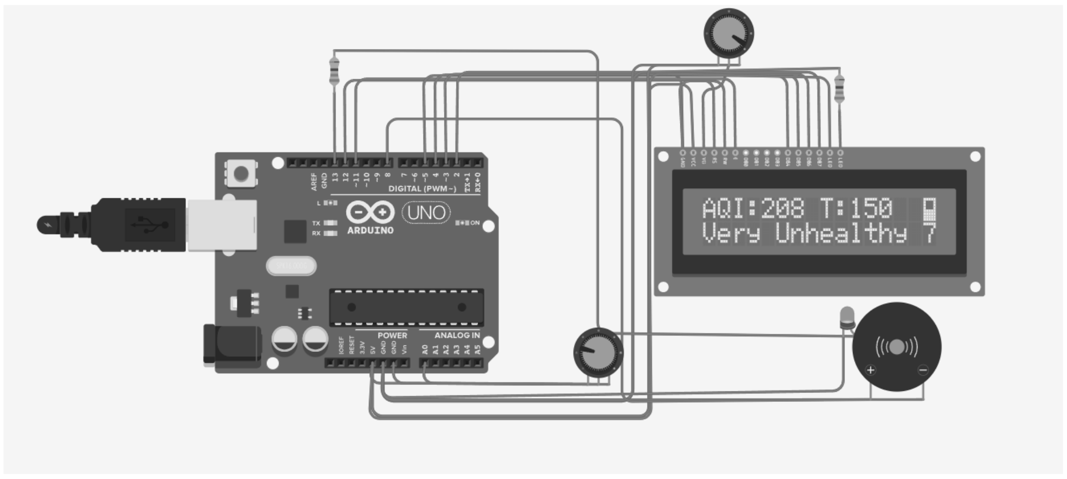
OUTPUT:

Air quality index below threshold:



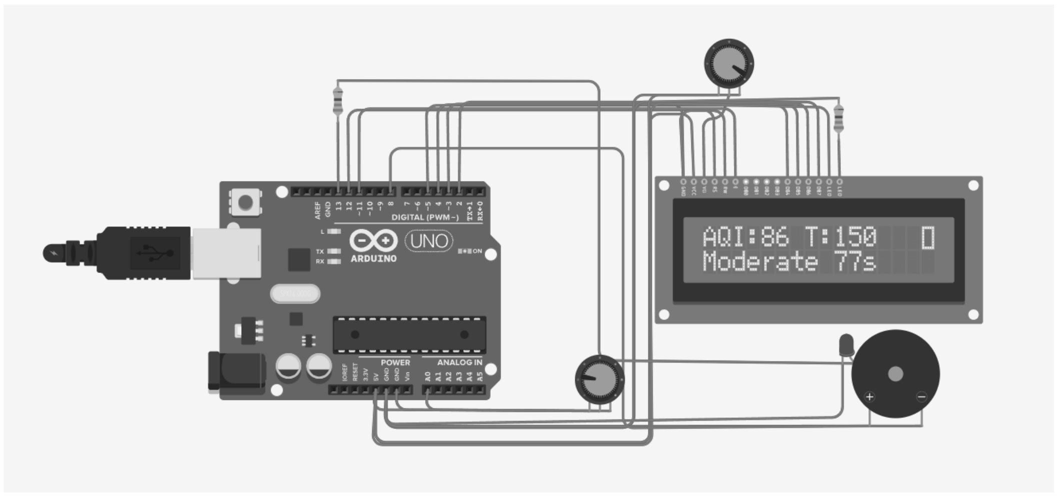
```
AQI:45 T:150
Good 4s
```

Air quality index above threshold (Air purifier on):



```
AQI:208 T:150
Very Unhealthy 7
```

Air quality index back to normal (air purifier off)



```
AQI:86 T:150
Moderate 77s
```

Conclusion: Implementation of Air quality monitor and smart air purifier was done successfully.