1. Komponen yang Digunakan

Berikut adalah komponen-komponen utama yang digunakan dalam proyek ini:

- **LCD I2C**: Digunakan untuk menampilkan informasi seperti jumlah ikan yang terdeteksi dan status sistem.
- **Keypad**: Digunakan untuk memasukkan data secara manual, seperti menambah atau mengurangi jumlah ikan.
- **Sensor Jarak (proximity infrared)**: Digunakan untuk mendeteksi ikan berdasarkan jarak.
- **Relay**: Digunakan untuk mengontrol pompa air, yang akan diaktifkan atau dinonaktifkan berdasarkan jumlah ikan yang terdeteksi.

---

2. Inisialisasi Komponen

Pada bagian awal kode, kita melakukan inisialisasi untuk LCD, keypad, dan sensor jarak:

```
LiquidCrystal_I2C lcd(0x3F, 20, 4); // LCD 20x4 dengan I2C

const int relayPin = A0; // Pin relay untuk pompa

const int proximityPins[] = {10, 11, 12, 13}; // Pin sensor jarak

int fishCount = 0; // Menyimpan jumlah ikan

String manualInput = ""; // Input manual dari keypad

bool isPaused = false; // Status pause

bool isManualEntry = false; // Status input manual
```

- LCD diinisialisasi dengan alamat I2C 0x3F dan ukuran 20x4.
- Relay pin diatur pada pin A0 untuk mengontrol pompa.
- Pin sensor jarak diatur pada pin digital 10 hingga 13.
- Variabel fishCount digunakan untuk menyimpan jumlah ikan yang terdeteksi.
- Variabel lainnya seperti isPaused dan isManualEntry digunakan untuk status sistem.

---

3. Fungsi Setup

Fungsi setup() dijalankan sekali ketika perangkat dinyalakan. Pada bagian ini, kita menginisialisasi LCD, pin relay, sensor, dan tampilan awal di LCD:

```
void setup() {
  lcd.begin();  // Inisialisasi LCD
  lcd.backlight(); // Menyalakan lampu latar LCD
  pinMode(relayPin, OUTPUT);  // Mengatur pin relay sebagai OUTPUT
  for (int i = 0; i < 4; i++) {
    pinMode(proximityPins[i], INPUT); // Mengatur pin sensor sebagai INPUT
  }
  Serial.begin(9600);  // Memulai komunikasi serial
  // Tampilan awal pada LCD
  lcd.setCursor(3, 0);
  lcd.print("ALAT PENDETEKSI");
  lcd.setCursor(0, 2);
  lcd.print("JUMLAH BENIH IKAN");
  delay(2000);
  lcd.clear();
}
```

- **LCD Initialization**: Menampilkan informasi awal mengenai sistem pada layar LCD.
- **Pin Configuration**: Menyusun pin untuk relay dan sensor.
- **Serial Communication**: Untuk memantau output melalui serial monitor.

---

4. Fungsi Loop

Fungsi loop() berisi proses yang berjalan terus-menerus selama sistem aktif. Di dalam fungsi ini, kita mengupdate jumlah ikan, memeriksa input dari keypad, dan mengendalikan pompa air.

```
void loop() {
  updateFishCount();  // Memperbarui jumlah ikan
  updateLCD(); // Memperbarui tampilan LCD
  char key = keypad.getKey();  // Mendeteksi tombol yang ditekan pada keypad
  if (key) {
    handleKeypadInput(key);  // Menangani input dari keypad
  }
  controlPump(); // Mengendalikan pompa air berdasarkan jumlah ikan
}
```

- **Update Fish Count**: Memperbarui jumlah ikan berdasarkan sensor.
- **Update LCD**: Menampilkan informasi terbaru pada layar LCD.
- **Keypad Input Handling**: Memproses input dari pengguna.
- **Control Pump**: Mengendalikan status pompa air.

---

5. Memperbarui Jumlah Ikan

Fungsi updateFishCount() bertugas untuk memperbarui jumlah ikan yang terdeteksi berdasarkan sensor jarak. Fungsi ini juga mengimplementasikan mekanisme **debouncing** untuk memastikan pembacaan yang stabil.

```
void updateFishCount() {
  static bool lastState[4] = {LOW, LOW, LOW, LOW};
  static unsigned long lastDebounceTime[4] = {0, 0, 0, 0};
  const unsigned long debounceDelay = 50;

  for (int i = 0; i < 4; i++) {
    bool currentState = digitalRead(proximityPins[i]) == HIGH;

    if (currentState != lastState[i] && currentState == HIGH) {
      if ((millis() - lastDebounceTime[i]) > debounceDelay) {
        fishCount--;
        if (fishCount < 0) fishCount = 0;
        lastDebounceTime[i] = millis();
        updateLCD();
      }
    }

    lastState[i] = currentState;
  }
}
```

- **Debouncing**: Menghindari pembacaan ganda yang tidak diinginkan dari sensor.

---

6. Memperbarui Tampilan LCD

Fungsi updateLCD() bertugas untuk memperbarui tampilan LCD berdasarkan jumlah ikan dan status pause. Fungsi ini juga menangani peralihan antar status (pause dan aktif).

```
void updateLCD() {
  static int lastFishCount = -1;
  static bool lastIsPaused = false;
  if (isPaused != lastIsPaused) {
    lcd.clear();
    if (isPaused) {
      lcd.setCursor(0, 0);
      lcd.print("Monitor PAUSED");
    }
    lastIsPaused = isPaused;
  }
  if (!isPaused && fishCount != lastFishCount) {
    lcd.setCursor(2, 0);
    lcd.print("AKUARIUM MONITOR");
    lcd.setCursor(0, 1);
    lcd.print("Jumlah Ikan: ");
    lcd.setCursor(12, 1);
    lcd.print("   ");
    lcd.setCursor(12, 1);
    lcd.print(fishCount);
    lastFishCount = fishCount;
  }
}
```

- **Pause State**: Menampilkan pesan "Monitor PAUSED" saat sistem dalam mode jeda.
- **Fish Count Display**: Menampilkan jumlah ikan yang terdeteksi.

7. Menangani Input Keypad

Fungsi handleKeypadInput() menangani input dari keypad, termasuk perintah untuk menambah atau mengurangi jumlah ikan, menjeda sistem, mereset jumlah ikan, dan memasukkan input manual.

```
void handleKeypadInput(char key) {
  if (isManualEntry) {
    if (key == '#') {
      if (manualInput.length() > 0) {
        fishCount = manualInput.toInt();
        updateLCD();
      }
      isManualEntry = false;
      manualInput = "";
      lcd.setCursor(0, 3);
      lcd.print("              ");
    } else if (key >= '0' && key <= '9') {
      if (manualInput.length() < 5) {
        manualInput += key;
        lcd.setCursor(0, 3);
        lcd.print("Input: ");
        lcd.print(manualInput);
      }
    }
  } else {
    switch (key) {
      case 'A':
        fishCount++;
        updateLCD();
```

```
      break;
    case 'B':
      if (fishCount > 0) {
        fishCount--;
        updateLCD();
      }
      break;
    case 'C':
      isPaused = !isPaused;
      updateLCD();
      break;
    case 'D':
      fishCount = 0;
      updateLCD();
      break;
    case '#':
      isManualEntry = true;
      manualInput = "";
      lcd.setCursor(0, 3);
      lcd.print("Input: ");
      break;
    default:
      break;
  }
 }
}
```

- **Manual Input**: Memungkinkan pengguna untuk memasukkan jumlah ikan secara manual.

- **Keypad Actions**: Tombol A, B, C, D memiliki fungsi untuk menambah, mengurangi, jeda, atau reset jumlah ikan.

---

8. Mengendalikan Pompa

Fungsi controlPump() mengendalikan status pompa berdasarkan jumlah ikan yang terdeteksi. Pompa dihidupkan jika ada ikan yang terdeteksi (jumlah ikan > 0) dan dimatikan jika tidak ada ikan.

```
void controlPump() {
  if (fishCount > 0) {
    digitalWrite(relayPin, HIGH);  // Menyalakan pompa
  } else {
    digitalWrite(relayPin, LOW);  // Mematikan pompa
  }
}
```