One-Time Alert Trigger Documentation

This document describes the one-time alert trigger functionality where alerts trigger only once and are then marked as "triggered" and no longer considered for future triggers.

Overview

The one-time trigger system ensures that:

- **Alerts trigger only once** when conditions are met
- **Status changes to "triggered"** after first trigger
- **No repeated triggers** for the same alert
- **Clear visual indicators** show trigger status
- **Database tracking** records trigger information

How It Works

1. **Initial State**

- Alert is created with status: `"enabled"`
- Alert count: `0`
- Alert is actively monitored

2. **Trigger Detection**

- System checks alert conditions every second
- When condition is met, alert triggers **once**
- Status changes to: `"triggered"`
- Alert count becomes: `1`

3. **Post-Trigger State**

- Alert status: `"triggered"`
- Alert count: `1` (never increases)
- Alert is **no longer monitored**
- Visual indicator shows "Triggered Once"

Database Changes

Status Updates

When an alert triggers:

```sql
UPDATE alerts
SET alert_count = 1,
    last_triggered_at = '2025-01-19T10:30:45.123456',
    last_triggered_price = 24500.50,
    status = 'triggered'
WHERE uuid = 'alert-uuid';
```

Monitoring Query

Only alerts with `alert_count = 0` are monitored:

```
SELECT * FROM alerts
WHERE status = 'enabled' AND alert_count = 0;
```

Visual Indicators

Status Badges

- **Green**: `"enabled"` - Alert is active and monitoring
- **Red**: `"triggered"` - Alert has triggered once
- **Gray**: `"inactive"` - Alert is disabled

Trigger Information

- **Badge**: "✅ Triggered Once"
- **Timestamp**: When the alert was triggered
- **Price**: Price at which it was triggered

User Interface Changes

Alert List Display

```
<div class="alert-status">
    <span class="status-badge triggered">triggered</span>
    <div class="trigger-info">
        <span class="trigger-badge">✅ Triggered Once</span>
        <br><small>At: 1/19/2025, 10:30:45 AM</small>
    </div>
</div>
```

CSS Styling

```
.status-badge.triggered {
    background: linear-gradient(135deg, #ff6b6b, #ee5a24);
    color: white;
}

.trigger-badge::before {
    content: "✅ ";
}
```

API Behavior

Check Triggers Endpoint

- Only checks alerts with `alert_count = 0`
- Once triggered, alerts are excluded from future checks
- Returns only newly triggered alerts

Stored Alerts Endpoint

- Shows all alerts including triggered ones
- Includes trigger information for triggered alerts
- Status field shows "triggered" for one-time triggers

Example Workflow

1. **Create Alert**

```
{
    "name": "NIFTY 50 Breakout",
    "lhs_exchange": "INDICES",
    "lhs_tradingsymbol": "NIFTY 50",
    "lhs_attribute": "LastTradedPrice",
    "operator": ">=",
    "rhs_type": "constant",
    "type": "simple",
    "rhs_constant": "24400"
}
```

2. **Alert State: Enabled**

- Status: `"enabled"`
- Count: `0`
- Monitoring: ✅ Active

3. **Price Hits Target**

- Current NIFTY 50: ₹24,500
- Target: ₹24,400
- Condition: `24500 >= 24400` = ✅ True

4. **Alert Triggers Once**

- Notification appears: "🚨 Alert Triggered!"

- Status changes to: `"triggered"`
- Count becomes: `1`
- Monitoring: ❌ Stopped

5. **Future Price Movements**

- Even if price goes to ₹25,000, alert won't trigger again
- Status remains: `"triggered"`
- Count remains: `1`

Benefits

1. **Prevents Spam**

- No repeated notifications for the same alert
- Clean user experience
- Reduces notification fatigue

2. **Clear Status**

- Easy to see which alerts have triggered
- Historical record of trigger events
- Clear distinction between active and triggered alerts

3. **Performance**

- Reduces monitoring load over time
- Fewer alerts to check as they trigger
- Efficient database queries

4. **User Control**

- Users can see trigger history
- Easy to identify which alerts have fired
- Can delete triggered alerts if needed

Testing

Test Script

```
python test_one_time_triggers.py
```

Manual Testing Steps

1. **Create Alert**: Set target close to current price
2. **Wait for Trigger**: Alert should trigger immediately
3. **Check Status**: Should show "triggered"

4. **Verify No Re-trigger**: Wait and confirm no more triggers
5. **Check Database**: Verify status and count updates

Expected Results

- ✅ Alert triggers once
- ✅ Status changes to "triggered"
- ✅ Count becomes 1
- ✅ No repeated triggers
- ✅ Visual indicators update

Comparison: Before vs After

Before (Multiple Triggers)

- Alert could trigger multiple times
- Status remained "enabled"
- Count kept increasing
- Continuous monitoring
- Potential notification spam

After (One-Time Triggers)

- Alert triggers only once
- Status changes to "triggered"
- Count becomes 1 and stays 1
- Monitoring stops after trigger
- Clean, predictable behavior

Use Cases

1. **Breakout Alerts**

- "NIFTY 50 Above ₹24,500"
- Triggers once when price breaks above level
- No repeated triggers on further price increases

2. **Support/Resistance**

- "NIFTY BANK Below ₹52,000"
- Triggers once when price breaks below level
- No repeated triggers on further price decreases

3. **Event-Based Alerts**

- "NIFTY 50 >= ₹25,000"
- Triggers once when milestone is reached
- Clear indication that event occurred

Migration Notes

Existing Alerts

- Alerts with `alert_count > 0` are considered already triggered
- Status will be updated to "triggered" on next trigger check
- No data loss during migration

Database Schema

- No schema changes required
- Existing fields used for new behavior
- Backward compatible

Best Practices

1. **Alert Creation**

- Set realistic targets
- Use descriptive names
- Consider one-time nature when setting targets

2. **Monitoring**

- Check triggered alerts periodically
- Clean up old triggered alerts
- Create new alerts for new conditions

3. **User Education**

- Explain one-time trigger behavior
- Show users how to identify triggered alerts
- Provide guidance on creating new alerts

The one-time trigger system provides a clean, predictable alert experience where each alert fires exactly once when its condition is met, then stops monitoring to prevent notification spam and provide clear status tracking.