Trading Alerts System

This document describes the trading alerts functionality that has been added to the Stock Panel application. The alerts system allows users to create, manage, and send trading alerts to the Kite 3 API.

Features

🔔 Alert Management

- **Create Alerts**: Set up price-based alerts for stocks and options
- **Bulk Create Alerts**: Add multiple alerts at once using a table interface
- **Edit Alerts**: Modify existing alert parameters
- **Delete Alerts**: Remove alerts you no longer need
- **Toggle Status**: Activate/deactivate alerts without deleting them
- **View All Alerts**: See all your alerts in a clean, organized interface

📊 Alert Types

- **Price Above**: Alert when price goes above a target value
- **Price Below**: Alert when price goes below a target value
- **Percentage Change**: Alert based on percentage change from a baseline

🎯 Instrument Types

- **Options**: Default alert type with underlying symbol, strike price, expiry, and option type (CALL/PUT)
- **Stocks**: Simple stock price alerts

🔗 Kite 3 API Integration

- **Automatic Sending**: Alerts are automatically sent to Kite 3 API when created
- **Configurable**: Easy setup with environment variables
- **Error Handling**: Graceful handling of API failures
- **Testing**: Built-in connection testing functionality

Setup

1. Environment Variables

To enable Kite 3 API integration, set the following environment variables:

```
# Kite 3 API Configuration
KITE_API_KEY=your_kite_api_key
KITE_API_SECRET=your_kite_api_secret
KITE_BASE_URL=https://api.kite.com/v3
```

2. Database Schema

The alerts system automatically creates the required database table when the application starts:

```
CREATE TABLE IF NOT EXISTS alerts (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    symbol TEXT NOT NULL,
    alert_type TEXT NOT NULL,
    target_value REAL NOT NULL,
    condition TEXT NOT NULL,
    message TEXT,
    is_active BOOLEAN DEFAULT 1,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    updated_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    user_id INTEGER NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users (id)
);
```

Usage

Accessing the Alerts Page

1. Login to the Stock Panel application
2. Navigate to the "Alerts" section from the sidebar
3. You'll see the alerts management interface

Creating an Alert

1. Click the "Add Alert" button
2. Fill in the required fields:
    - **Symbol**: The option symbol (e.g., RELIANCE24JAN2500CE) or stock symbol
    - **Instrument Type**: Choose between Option (default) or Stock
    - **Underlying Symbol**: For options, the underlying stock (e.g., RELIANCE)
    - **Option Type**: For options, choose Call or Put
    - **Strike Price**: For options, the strike price
    - **Expiry Date**: For options, the expiry date
    - **Alert Type**: Choose from Price Above, Price Below, or Percentage Change
    - **Target Value**: The price or percentage threshold
    - **Condition**: The comparison operator (>, <, >=, <=, ==)
    - **Message**: Optional custom message for the alert
3. Click "Create Alert"

Bulk Creating Alerts

The bulk alert creation interface is available directly on the alerts page:

1. Scroll to the "Bulk Create Alerts" section
2. Use the table interface to add multiple alerts:

    - Click "Add Row" to add a new alert row
    - Fill in the fields for each row
    - Use "Load Sample" to see example data
    - Use "Clear All" to reset the table
3. Click "Create All Alerts" to create all alerts at once

Managing Alerts

- **Edit**: Click the "Edit" button on any alert card
- **Toggle**: Use the "Activate"/"Deactivate" button to control alert status
- **Delete**: Click the "Delete" button to remove an alert

API Endpoints

Protected Endpoints (Require Authentication)

Method	Endpoint	Description
GET	`/alerts`	Get all alerts for the current user
POST	`/alerts`	Create a new alert
PUT	`/alerts?id={id}`	Update an existing alert
DELETE	`/alerts?id={id}`	Delete an alert
PATCH	`/alerts/toggle?id={id}`	Toggle alert active status
POST	`/alerts/test-kite`	Test Kite 3 API connection

Request/Response Examples

Create Alert Request (Option)

```
{
  "symbol": "RELIANCE24JAN2500CE",
  "underlying_symbol": "RELIANCE",
  "option_type": "CALL",
  "strike_price": 2500.0,
  "expiry": "2024-01-25",
  "alert_type": "PRICE_ABOVE",
  "target_value": 50.0,
  "condition": ">",
  "message": "RELIANCE Call above 50"
}
```

Create Alert Request (Stock)

```
{
  "symbol": "RELIANCE",
  "alert_type": "PRICE_ABOVE",
  "target_value": 2500.0,
  "condition": ">",
  "message": "RELIANCE has crossed 2500"
}
```

Create Alert Response

```
{
  "success": true,
  "message": "Alert created successfully",
  "alert": {
    "id": 1,
    "symbol": "RELIANCE24JAN2500CE",
    "underlying_symbol": "RELIANCE",
    "option_type": "CALL",
    "strike_price": 2500.0,
    "expiry": "2024-01-25",
    "alert_type": "PRICE_ABOVE",
    "target_value": 50.0,
    "condition": ">",
    "message": "RELIANCE Call above 50",
    "is_active": true,
    "created_at": "2024-01-15T10:30:00Z",
    "updated_at": "2024-01-15T10:30:00Z",
    "user_id": 1
  }
}
```

Testing

Test Page

Access the test page at `/test-alerts` to verify the alerts system:

1. **Test Kite API Connection**: Verify Kite 3 API is properly configured
2. **Create Test Alert**: Create a sample alert to test the system
3. **List All Alerts**: View all alerts for the current user
4. **System Information**: Check system configuration and status

Manual Testing

1. Start the application: `go run cmd/stock-panel/main.go`
2. Login with any test account (admin/password123, demo/demo123, vinay/password123)
3. Navigate to `/test-alerts` or `/web/alerts/`

4. Test the various alert functions

Kite 3 API Integration

Alert Payload Structure

When an alert is created, it's automatically sent to the Kite 3 API with this structure:

```
{
  "symbol": "RELIANCE",
  "alert_type": "PRICE_ABOVE",
  "target_value": 2500.0,
  "condition": ">",
  "message": "RELIANCE has crossed 2500",
  "timestamp": "2024-01-15T10:30:00Z",
  "user_id": 1
}
```

API Headers

The Kite API requests include these headers:

- `Content-Type: application/json`
- `X-API-Key: {your_api_key}`
- `X-API-Secret: {your_api_secret}`

Error Handling

- If Kite API is not configured, alerts are still saved locally
- If Kite API fails, the alert creation continues but logs a warning
- Connection failures don't prevent the application from working

Security

- All alert endpoints require authentication
- Users can only access their own alerts
- User ID is automatically set from the authentication context
- SQL injection protection through parameterized queries

File Structure

```
internal/
├── models/
│   └── alert.go          # Alert data models
├── handlers/
│   └── alerts.go         # Alert HTTP handlers
├── kite/
│   └── kite.go           # Kite 3 API integration
```

```
└── db/
    └── db.go             # Database schema (updated)

web/
├── pages/
│   └── alerts.html       # Alerts management page
└── js/
    └── alerts.js         # Frontend alerts functionality

test-alerts.html          # Test page for alerts system
```

Troubleshooting

Common Issues

1. **Kite API Connection Fails**
    - Check environment variables are set correctly
    - Verify API credentials are valid
    - Ensure network connectivity to Kite API
2. **Alerts Not Saving**
    - Check database permissions
    - Verify user authentication
    - Check server logs for errors
3. **Frontend Not Loading**
    - Ensure you're logged in
    - Check browser console for JavaScript errors
    - Verify all static files are accessible

Logs

The application logs all alert operations:

- Alert creation, updates, and deletions
- Kite API communication attempts
- Authentication and authorization events

Future Enhancements

- **Real-time Price Monitoring**: Automatic price checking against alerts
- **Email Notifications**: Send alert notifications via email
- **Webhook Support**: Send alerts to external webhooks
- **Alert Templates**: Predefined alert templates for common scenarios
- **Bulk Operations**: Create multiple alerts at once
- **Alert History**: Track when alerts were triggered

- **Mobile Notifications**: Push notifications for triggered alerts