Database API Documentation

This document describes the database functionality for storing and retrieving alert responses from the KITE API.

Overview

The application now includes SQLite database functionality to store all alert responses from the KITE API. This allows you to:

- Store alert responses automatically when creating alerts
- Retrieve all stored alerts
- Get specific alerts by UUID
- Track alert history and status

Database Schema

Alerts Table

The `alerts` table stores all alert information with the following structure:

Column	Type	Description
`id`	INTEGER	Primary key (auto-increment)
`uuid`	TEXT	Unique alert identifier from KITE API
`name`	TEXT	Alert name
`user_id`	TEXT	User ID from KITE API
`lhs_exchange`	TEXT	Left-hand side exchange
`lhs_tradingsymbol`	TEXT	Left-hand side trading symbol
`lhs_attribute`	TEXT	Left-hand side attribute (e.g., LastTradedPrice)
`operator`	TEXT	Comparison operator (>=, <=, >, <, ==, !=)
`rhs_type`	TEXT	Right-hand side type (constant/variable)
`rhs_constant`	REAL	Right-hand side constant value
`rhs_exchange`	TEXT	Right-hand side exchange (if variable)
`rhs_tradingsymbol`	TEXT	Right-hand side trading symbol (if variable)
`rhs_attribute`	TEXT	Right-hand side attribute (if variable)
`type`	TEXT	Alert type (simple, etc.)
`status`	TEXT	Alert status (enabled, disabled, etc.)

Column	Type	Description
`alert_count`	INTEGER	Number of times alert has triggered
`disabled_reason`	TEXT	Reason for disabling (if applicable)
`created_at`	TEXT	Alert creation timestamp from KITE
`updated_at`	TEXT	Alert last update timestamp from KITE
`stored_at`	TEXT	Local storage timestamp
`kite_response`	TEXT	Full JSON response from KITE API

API Endpoints

1. Create Alert (Enhanced)

Endpoint: `POST /alerts/create`

Description: Creates a new alert, sends it to KITE API, and automatically stores the response in the database.

Request: Same as before - no changes to the request format.

Response: Enhanced with database storage confirmation.

```
{
    "message": "Alert created successfully",
    "success": true,
    "response": {
        "data": {
            "uuid": "b88f3994-4d51-4266-b7d4-85ac2e2f7212",
            "name": "NIFTY 50 Alert Test",
            "user_id": "YL5749",
            "status": "enabled",
            "created_at": "2025-09-19 13:40:37",
            "updated_at": "2025-09-19 13:40:37",
            "alert_count": 0,
            "disabled_reason": "",
            "lhs_exchange": "INDICES",
            "lhs_tradingsymbol": "NIFTY 50",
            "lhs_attribute": "LastTradedPrice",
            "operator": "<=",
            "rhs_type": "constant",
            "rhs_constant": 23533,
            "type": "simple"
        },
        "status": "success"
    }
}
```

2. Get All Stored Alerts

Endpoint: `GET /alerts/stored`

Description: Retrieves all alerts stored in the local database.

Response:

```
{
    "alerts": [
        {
            "uuid": "b88f3994-4d51-4266-b7d4-85ac2e2f7212",
            "name": "NIFTY 50 Alert Test",
            "user_id": "YL5749",
            "status": "enabled",
            "created_at": "2025-09-19 13:40:37",
            "updated_at": "2025-09-19 13:40:37",
            "stored_at": "2025-01-19T10:30:45.123456",
            "alert_count": 0,
            "disabled_reason": "",
            "lhs_exchange": "INDICES",
            "lhs_tradingsymbol": "NIFTY 50",
            "lhs_attribute": "LastTradedPrice",
            "operator": "<=",
            "rhs_type": "constant",
            "rhs_constant": 23533,
            "type": "simple"
        }
    ],
    "count": 1,
    "success": true
}
```

3. Get Alert by UUID

Endpoint: `GET /alerts/stored/<uuid>`

Description: Retrieves a specific alert by its UUID, including the full KITE API response.

Response:

```
{
    "alert": {
        "uuid": "b88f3994-4d51-4266-b7d4-85ac2e2f7212",
        "name": "NIFTY 50 Alert Test",
        "user_id": "YL5749",
        "status": "enabled",
        "created_at": "2025-09-19 13:40:37",
        "updated_at": "2025-09-19 13:40:37",
        "stored_at": "2025-01-19T10:30:45.123456",
        "alert_count": 0,
        "disabled_reason": "",
        "lhs_exchange": "INDICES",
        "lhs_tradingsymbol": "NIFTY 50",
        "lhs_attribute": "LastTradedPrice",
        "operator": "<=",
        "rhs_type": "constant",
        "rhs_constant": 23533,
        "type": "simple",
        "kite_response": {
            "data": {
                "uuid": "b88f3994-4d51-4266-b7d4-85ac2e2f7212",
                "name": "NIFTY 50 Alert Test",
                "user_id": "YL5749",
                "status": "enabled",
                "created_at": "2025-09-19 13:40:37",
                "updated_at": "2025-09-19 13:40:37",
                "alert_count": 0,
                "disabled_reason": "",
                "lhs_exchange": "INDICES",
                "lhs_tradingsymbol": "NIFTY 50",
                "lhs_attribute": "LastTradedPrice",
                "operator": "<=",
                "rhs_type": "constant",
                "rhs_constant": 23533,
                "type": "simple"
            },
            "status": "success"
        }
    },
    "success": true
}
```

Example Usage

Create Alert and Store in Database

```
curl -X POST http://localhost:5001/alerts/create \
    -H "Content-Type: application/json" \
    -d '{
        "name": "NIFTY 50 Database Test",
        "lhs_exchange": "INDICES",
        "lhs_tradingsymbol": "NIFTY 50",
        "lhs_attribute": "LastTradedPrice",
        "operator": ">=",
        "rhs_type": "constant",
        "type": "simple",
        "rhs_constant": "27000"
    }'
```

Get All Stored Alerts

```
curl -X GET http://localhost:5001/alerts/stored
```

Get Specific Alert by UUID

```
curl -X GET http://localhost:5001/alerts/stored/b88f3994-4d51-4266-b7d4-
85ac2e2f7212
```

Using Python requests

```python
import requests
import json

# Create alert (automatically stored in database)
alert_data = {
    "name": "NIFTY 50 Database Test",
    "lhs_exchange": "INDICES",
    "lhs_tradingsymbol": "NIFTY 50",
    "lhs_attribute": "LastTradedPrice",
    "operator": ">=",
    "rhs_type": "constant",
    "type": "simple",
    "rhs_constant": "27000"
}

response = requests.post(
    "http://localhost:5001/alerts/create",
    json=alert_data,
    headers={'Content-Type': 'application/json'}
)

if response.status_code == 200:
    result = response.json()
    uuid = result['response']['data']['uuid']
    print(f"Alert created with UUID: {uuid}")

    # Get all stored alerts
    stored_response =
requests.get("http://localhost:5001/alerts/stored")
    alerts = stored_response.json()['alerts']
    print(f"Total stored alerts: {len(alerts)}")

    # Get specific alert
    specific_response =
requests.get(f"http://localhost:5001/alerts/stored/{uuid}")
```

```
    alert = specific_response.json()['alert']
    print(f"Alert status: {alert['status']}")
```

Database File

The database is stored as `alerts.db` in the application root directory. This is a SQLite database file that can be:

- Backed up by copying the file
- Inspected using SQLite tools
- Migrated to other systems

Testing

Run the database functionality test:

```
python test_database_functionality.py
```

This will test:

- Alert creation and storage
- Retrieving all stored alerts
- Getting specific alerts by UUID
- Database schema validation

Features

Automatic Storage

- All successful alert creations are automatically stored in the database
- No additional API calls needed - storage happens transparently

Complete Data Preservation

- Full KITE API response is stored as JSON
- All alert parameters are stored in structured format
- Timestamps for both KITE creation and local storage

Easy Retrieval

- Get all alerts with a single API call
- Retrieve specific alerts by UUID
- Access both structured data and raw KITE response

Data Integrity

- UUID-based uniqueness prevents duplicates

- INSERT OR REPLACE ensures data consistency
- Proper error handling for database operations

Notes

- Authentication is required for all database endpoints
- The database is automatically initialized on application startup
- All timestamps are stored in ISO format
- The `kite_response` field contains the complete JSON response from KITE API