Watchlist Alert System Documentation

Overview

The Watchlist Alert System provides real-time monitoring of blockchain wallets with **dual notification delivery**: in-app alerts and Telegram messages. When a watchlisted wallet makes a trade, users receive instant notifications through both channels.

Architecture

Components

1. Database Models (`/app/prisma/schema.prisma`)

```
model WatchlistItem {
  id            String   @id @default(cuid())
  userId        String
  address       String
  chain         String
  chainId       String?
  label         String?
  tokenAddress  String?
  tokenSymbol   String?
  lastChecked   DateTime @default(now())
  createdAt     DateTime @default(now())
  user          User     @relation(fields: [userId], references: [id], onDelete: Cas-
cade)

  @@unique([userId, address, chain])
  @@index([userId])
}

model TransactionAlert {
  id              String    @id @default(cuid())
  userId          String
  walletAddress   String
  chain           String
  transactionHash String
  fromAddress     String?
  toAddress       String?
  value           String?
  tokenAddress    String?
  tokenSymbol     String?
  tokenAmount     String?
  type            String    // sent, received, swap, contract
  isRead          Boolean   @default(false)
  notifiedAt      DateTime  @default(now())
  createdAt       DateTime  @default(now())
  user            User      @relation(fields: [userId], references: [id], onDelete: Ca
scade)

  @@unique([userId, transactionHash])
  @@index([userId, isRead])
}
```

2. Monitoring Daemon (`/scripts/monitor-watchlist.ts`)

- **Schedule**: Runs every 1 hour (3600 seconds)
- **Function**: Calls `/api/watchlist/check` endpoint
- **Status**: ACTIVE (automatically started)
- **Logs**: Stored in `/home/ubuntu/watchlist_logs/`

3. Watchlist Check API (`/app/api/watchlist/check/route.ts`)

Endpoint: `POST /api/watchlist/check`

Process Flow:
1. **Cleanup Phase**: Remove watchlist items for expired trial users
2. **Fetch Phase**: Get all active watchlist items from database
3. **Scan Phase**: For each wallet:
- Query blockchain APIs (Alchemy, Moralis, Etherscan for EVM; Helius for Solana)
- Fetch last 10 transactions
- Filter for transactions newer than `lastChecked` timestamp
- For token-specific watchlist items, filter by token address
4. **Alert Phase**: For each new transaction:
- Create `TransactionAlert` record in database
- Send Telegram notification (if user has linked Telegram)
5. **Update Phase**: Update `lastChecked` timestamp for each wallet

Response:

```
{
  "success": true,
  "walletsChecked": 15,
  "alertsCreated": 3,
  "results": [
    {
      "address": "0x1234...",
      "chain": "ethereum",
      "newTransactions": 2
    }
  ]
}
```

4. Telegram Client (`/lib/telegram-client.ts`)

Class: `TelegramClient`
Bot Token: Stored in `/home/ubuntu/.config/abacusai_auth_secrets.json`

Key Methods:
- `sendWalletTransactionAlert()` : Sends formatted transaction alert to user
- `notifyWalletTransaction()` : Helper function that fetches user's Telegram chat ID and sends alert

Message Format:

```
📤 SENT Transaction

🔗 Chain: ETHEREUM
💼 Wallet: `0x8A9E89...c7D87a`
💰 Amount: 1.5 USDC

🔍 View on Explorer
```

5. In-App Alerts Component (`/components/wallet-tracker/transaction-alerts.tsx`)

Location: Mobile Header (top-right corner)
Features:
- Bell icon with unread badge count
- Dropdown menu showing last 50 alerts
- Auto-refreshes every 30 seconds
- Click alert to mark as read
- “Clear Read” button to remove old alerts
- Direct links to blockchain explorers

Alert Display:
- **Unread**: Green highlighted background
- **Read**: Normal background
- **Info**: Transaction type, chain, token details, timestamp
- **Actions**: Mark as read, view on explorer, clear

6. Alerts Management API (`/app/api/watchlist/alerts/route.ts`)

Endpoints:

- **GET** `/api/watchlist/alerts`
- Returns user’s alerts (last 50)
- Returns unread count
- Requires authentication
- **PATCH** `/api/watchlist/alerts`
- Marks alerts as read
- Body: `{ "alertIds": ["alert1", "alert2"] }`
- **DELETE** `/api/watchlist/alerts`
- Deletes all read alerts for user
- No body required

How It Works: Complete Flow

Setup Phase (One-Time)

1. User navigates to **Wallet Tracker** page (`/wallet-tracker`)
2. User adds wallet addresses to watchlist with:
   - Wallet address
   - Blockchain (Ethereum, Base, BNB, Polygon, Solana, etc.)

   - Optional: Specific token to monitor
   - Optional: Label for identification
3. User connects Telegram account (optional, for Telegram notifications):
   - Visit `/settings` page
   - Link Telegram username and chat ID
   - Or use Telegram bot to auto-link

Monitoring Phase (Automated)

1. **Daemon Trigger** (every 1 hour):
   - Scheduled task wakes up
   - Executes `/scripts/monitor-watchlist.ts`
2. **API Call**:
   - Script calls `POST /api/watchlist/check`
   - Backend fetches all active watchlist items
3. **Blockchain Scanning**:
   - For each wallet, queries blockchain APIs:
     - **EVM Chains**: Alchemy → Moralis → Etherscan (fallback)
     - **Solana**: Helius RPC
     - Retrieves last 10 transactions
     - Compares timestamps with `lastChecked`
4. **Transaction Detection**:
   - Filters for new transactions since last check
   - For token-specific watchlists, filters by token address
   - Determines transaction type:
     - **Sent**: Wallet is sender
     - **Received**: Wallet is receiver
     - **Contract**: Smart contract interaction
     - **Swap**: Token swap detected
5. **Alert Creation**:
   - Creates `TransactionAlert` record in database
   - Stores transaction details:
     - Hash, from/to addresses, value
     - Token symbol, token amount (if applicable)
     - Chain, type, timestamp
6. **Notification Dispatch**:
   - **In-App**: Alert automatically appears in database
   - **Telegram**: Sends formatted message to user's chat
   - User sees bell icon badge update in real-time

User Interaction Phase

1. **In-App**:
   - Bell icon shows unread count
   - Click to view alert dropdown
   - Click alert to mark as read

- Click explorer link to view on blockchain
- Clear read alerts

2. **Telegram**:
- Receives instant message with transaction details
- Click explorer link in message
- No action needed in app

Supported Blockchains

Blockchain	Chain ID	API Providers	Explorer
Ethereum	`0x1`	Alchemy, Moralis, Etherscan	etherscan.io
Base	`0x2105`	Alchemy, Moralis, Etherscan	basescan.org
BNB Chain	`0x38`	Moralis, Etherscan	bscscan.com
Polygon	`0x89`	Alchemy, Moralis, Etherscan	polygonscan.com
Optimism	`0xa`	Alchemy, Moralis, Etherscan	optimist-ic.etherscan.io
Arbitrum	`0xa4b1`	Alchemy, Moralis, Etherscan	arbiscan.io
Solana	N/A	Helius RPC	solscan.io

API Rate Limits & Redundancy

Multi-Provider Fallback Strategy

1. **Primary**: Alchemy (highest reliability, lowest latency)
2. **Fallback 1**: Moralis (comprehensive multi-chain support)
3. **Fallback 2**: Etherscan (historical data, rate-limited)

Rate Limits (Per Minute)

- **Alchemy**: 330 requests/second (free tier)
- **Moralis**: 1,500 requests/minute
- **Etherscan**: 5 requests/second (free tier)

The system automatically switches to backup providers if primary fails.

Premium vs Free Features

Feature	Free (Trial)	Premium
Watchlist wallets	3	Unlimited
Transaction alerts	✅	✅
Telegram notifications	✅	✅
Alert history	7 days	90 days
Monitoring frequency	1 hour	1 hour
Token-specific alerts	❌	✅
Multi-chain support	Limited	All chains

Environment Variables Required

```
# Blockchain APIs
ALCHEMY_API_KEY=your_alchemy_key
MORALIS_API_KEY=your_moralis_key
ETHERSCAN_API_KEY=your_etherscan_key
HELIUS_API_KEY=your_helius_key

# Database
DATABASE_URL=your_postgres_connection_string

# Telegram
# Stored in /home/ubuntu/.config/abacusai_auth_secrets.json
telegram.secrets.bot_token.value=your_telegram_bot_token

# App URL
NEXT_PUBLIC_APP_URL=https://defidashtracker.com
```

Daemon Task Details

Task Name: Watchlist Wallet Monitor
Type: Scheduled
Interval: 3600 seconds (1 hour)
Status: ACTIVE
Next Run: Auto-calculated by scheduler
Script: `/home/ubuntu/smart_money_tracker/app/scripts/monitor-watchlist.ts`
Logs: `/home/ubuntu/watchlist_logs/monitor_<timestamp>.md`

Monitoring the Daemon

```
# View logs
ls -la /home/ubuntu/watchlist_logs/

# Read latest log
cat /home/ubuntu/watchlist_logs/monitor_latest.md

# Check task status
# (Use the daemon management interface)
```

Testing the Alert System

1. Manual Trigger

```
cd /home/ubuntu/smart_money_tracker/app
npx ts-node scripts/monitor-watchlist.ts
```

2. Test with Sample Wallet

```
curl -X POST http://localhost:3000/api/watchlist/check \
  -H "Content-Type: application/json"
```

3. Verify Alerts

```
# In-app: Visit /wallet-tracker and check bell icon
# Telegram: Check bot messages
# Database: Query TransactionAlert table
```

Troubleshooting

No Alerts Appearing

1. **Check Daemon Status**: Ensure scheduled task is ACTIVE
2. **Verify API Keys**: Test blockchain API connections
3. **Check lastChecked**: Ensure timestamps are not in future
4. **Review Logs**: Check `/home/ubuntu/watchlist_logs/`

Telegram Not Working

1. **Verify Bot Token**: Check `/home/ubuntu/.config/abacusai_auth_secrets.json`
2. **Check Chat ID**: User must have linked Telegram in settings
3. **Test Bot**: Send `/start` to bot to verify it's running
4. **Review Telegram Client**: Check `telegram-client.ts` for errors

Missing Transactions

1. **Check Blockchain APIs**: Verify Alchemy, Moralis, Etherscan are working
2. **Review Filters**: Ensure token-specific filters are correct
3. **Check Frequency**: Daemon runs every 1 hour, fast transactions may be missed
4. **API Rate Limits**: Check if any API is rate-limited

Future Enhancements

- [ ] Real-time WebSocket monitoring (reduce 1-hour delay)
- [ ] Advanced filters (transaction value thresholds)
- [ ] Email notifications
- [ ] Discord notifications
- [ ] Custom alert frequencies (premium users)
- [ ] NFT transaction alerts
- [ ] DeFi protocol-specific alerts
- [ ] Whale movement predictions

API Documentation

Add Wallet to Watchlist

```
POST /api/watchlist
Content-Type: application/json

{
  "address": "0x742d35Cc6634C0532925a3b844Bc9e7595f0bEb",
  "chain": "ethereum",
  "label": "Whale Wallet #1",
  "tokenAddress": "0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48", // Optional: USDC
  "tokenSymbol": "USDC" // Optional
}
```

Get Watchlist

```
GET /api/watchlist
```

Remove from Watchlist

```
DELETE /api/watchlist?id=watchlist_item_id
```

Trigger Manual Check

```
POST /api/watchlist/check
```

Get Alerts

```
GET /api/watchlist/alerts
```

Mark Alerts as Read

```
PATCH /api/watchlist/alerts
Content-Type: application/json

{
  "alertIds": ["alert1", "alert2", "alert3"]
}
```

Summary

The Watchlist Alert System provides comprehensive, automated monitoring of blockchain wallets with **dual notification delivery**:

✅ **In-App Alerts**: Bell icon with dropdown, auto-refreshing, mark as read
✅ **Telegram Notifications**: Instant messages with transaction details
✅ **Multi-Chain Support**: Ethereum, Base, BNB, Polygon, Arbitrum, Optimism, Solana
✅ **Automated Monitoring**: Daemon runs every 1 hour
✅ **Triple Redundancy**: Alchemy → Moralis → Etherscan fallback
✅ **Token-Specific Tracking**: Monitor specific tokens only (premium)
✅ **Transaction History**: Full alert history with blockchain explorer links

Users simply add wallets to their watchlist, and the system handles everything automatically.