Market Sentiment Feature Technical Documentation

1. Feature Overview

The **Institutional Sentiment** feature analyzes price trends across two timeframes to determine bullish/bearish market sentiment for NEPSE stocks. It combines:

- **Higher timeframe (1W)**: Identifies long-term trend
- **Lower timeframe (1D)**: Captures short-term momentum
- **EMA filters (5, 10, 20 periods)**: Confirms trend strength

2. Sentiment Calculation Logic

A. Conditions for Bullish Sentiment

Timeframe	Condition
1W (Higher)	Current price > 5EMA > 10EMA > 20EMA
1D (Lower)	Current price > 5EMA > 10EMA > 20EMA

B. Conditions for Bearish Sentiment

Timeframe	Condition
1W (Higher)	Current price < 5EMA < 10EMA < 20EMA
1D (Lower)	Current price < 5EMA < 10EMA < 20EMA

C. Sentiment Score Formula

text

```
Bullish Score = (Higher TF Confirmation * 0.6) + (Lower TF Confirmation * 0.4)
```

- Where confirmation = 1 if all EMAs align, else 0
- Final score converted to percentage (e.g., 72% = Strong Bullish)

3. Data Requirements

Input Data (Per Symbol)

Field	Description	Example
`symbol`	Stock/Forex pair	NIC, AAPL
`price_1w`	Weekly close price	1200.50
`ema5_1w`	5-period weekly EMA	1180.20
`ema10_1w`	10-period weekly EMA	1150.75
`ema20_1w`	20-period weekly EMA	1100.30
`price_1d`	Daily close price	1215.25
`ema5_1d`	5-period daily EMA	1205.10
`ema10_1d`	10-period daily EMA	1190.40
`ema20_1d`	20-period daily EMA	1175.80

4. Technical Implementation

A. Backend (Python)

python

```python
def calculate_sentiment(symbol_data):
    # Higher TF (1W) Check
    htf_bullish = (symbol_data['price_1w'] > symbol_data['ema5_1w'] >
                   symbol_data['ema10_1w'] > symbol_data['ema20_1w'])
    htf_bearish = (symbol_data['price_1w'] < symbol_data['ema5_1w'] <
                   symbol_data['ema10_1w'] < symbol_data['ema20_1w'])

    # Lower TF (1D) Check
    ltf_bullish = (symbol_data['price_1d'] > symbol_data['ema5_1d'] >
                   symbol_data['ema10_1d'] > symbol_data['ema20_1d'])
    ltf_bearish = (symbol_data['price_1d'] < symbol_data['ema5_1d'] <
                   symbol_data['ema10_1d'] < symbol_data['ema20_1d'])

    # Score Calculation
    score = 0
    if htf_bullish and ltf_bullish:
        score = 0.6 * 1 + 0.4 * 1  # Max bullish
    elif htf_bullish or ltf_bullish:
        score = 0.6 * htf_bullish + 0.4 * ltf_bullish

    return {
        'symbol': symbol_data['symbol'],
        'score': int(score * 100),
        'trend': 'Bullish' if score >= 50 else 'Bearish'
    }
```

B. Database Schema

sql

```
CREATE TABLE sentiment_data (
    symbol VARCHAR(10) PRIMARY KEY,
    price_1w DECIMAL(10,2),
    ema5_1w DECIMAL(10,2),
    ema10_1w DECIMAL(10,2),
    ema20_1w DECIMAL(10,2),
    price_1d DECIMAL(10,2),
    ema5_1d DECIMAL(10,2),
    ema10_1d DECIMAL(10,2),
    ema20_1d DECIMAL(10,2),
    last_updated TIMESTAMP
);
```

5. Frontend Display

UI Components

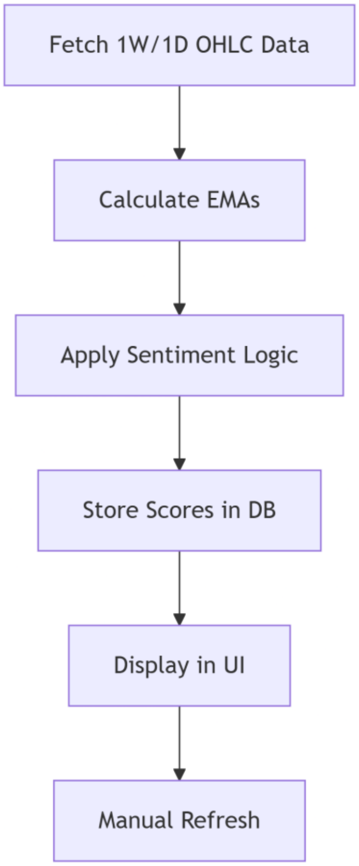
1. **Search Bar**
   - Typeahead symbol search
2. **Sentiment Table**
   - Columns: Symbol, Intraday (1D), Daily (1W)
   - Color coding:
     - Green (≥70% Bullish)
     - Red (≤30% Bullish)
     - Yellow (Neutral)
3. **Refresh Button**
   - Manual data reload

Example Response

json

```
{
    "AAPL": {"intraday": 72, "daily": 28},
    "NIC": {"intraday": 85, "daily": 15}
}
```

6. Workflow Diagram



7. Testing Scenarios

Test Case	Expected Result
Price > All EMAs (Both TFs)	100% Bullish
Price < All EMAs (Both TFs)	0% Bullish
Mixed Alignment	30-70% Score
Missing Data	"N/A" Display

Next Steps:

1. Implement EMA calculation microservice
2. Build real-time data pipeline
3. Optimize for 100+ symbols