**Market Magician API Documentation**

**Overview**

The API provides endpoints to predict stock prices using machine learning models (LSTMs). It either loads a pre-trained model for a ticker or trains one on-demand if no model exists.

**Endpoint: POST /api/predict/**

Description: Predicts the stock price for a given ticker using an LSTM model. If no trained model exists, it trains the model dynamically and saves it for future predictions.

**Method:** POST

**Request Body** (JSON):

**Code:**

{  
 "ticker": "AAPL"  
}

ticker (string, required): The stock ticker symbol. Defaults to "AAPL" if not provided.

**Response Format**

Success Response:

**Code:**

{  
 "ticker": "AAPL",  
 "predicted\_risk": "Moderate",  
 "classification": "Moderate",  
 "low\_threshold": 140.25,  
 "high\_threshold": 160.75,  
 "predicted\_price": 155.50  
}

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| ticker | string | The stock ticker symbol. |
| predicted\_risk | string | The risk classification (e.g., Low, Moderate). |
| classification | string | Same as predicted\_risk (for UI alignment). |
| low\_threshold | float | Calculated low price threshold for risk analysis. |
| high\_threshold | float | Calculated high price threshold for risk analysis. |
| predicted\_price | float, | Predicted stock price rounded to 2 decimal places. |

**Error Response:**

|  |  |  |
| --- | --- | --- |
| Code | Message | Example |
| 404 | Data not available for ticker | { "error": "Data not available for AAPL." } |
| 400 | Invalid request format | { "error": "Invalid JSON format." } |
| 500 | Internal server error (during training) | { "error": "An unexpected error occurred." } |

**Example Axios Client Code (Frontend)**

import axios from 'axios';  
  
const fetchPrediction = async (ticker) => {  
 try {  
 const response = await axios.post('http://localhost:8000/api/predict/', {  
 ticker: ticker || 'AAPL',  
 });  
 console.log('Prediction Response:', response.data);  
 return response.data;  
 } catch (error) {  
 console.error('Error fetching prediction:', error.response?.data || error.message);  
 }  
};  
  
// Example usage:  
fetchPrediction('AAPL');

**Tech Stack**

* Backend: Django REST Framework (DRF)
* Frontend: React with Axios for API communication
* Machine Learning: LSTM neural networks using Keras/TensorFlow
* Data Source: Yahoo Finance (yfinance library)
* Model Management: Custom functions to save/load trained models

**Key Notes**

1. Model Training Logic:

* If no model exists for the given ticker, the backend:
* Downloads stock price data using Yahoo Finance.
* Preprocesses data using MinMaxScaler.
* Creates and trains an LSTM model.
* Saves the trained model for future requests.

1. Prediction:

* Predicts stock price based on the most recent 60 days' closing prices.

1. Risk Classification:

* Risk is classified based on thresholds (33rd and 66th percentile of closing prices).

**Why Axios and Node.js?**

Axios:

* Used to send HTTP requests from the frontend to the backend.
* Simplifies error handling and supports JSON payloads.

Node.js:

* Required for managing the React frontend environment using tools like npm or yarn.
* Not part of the backend in this setup since Django handles API requests.

**How It All Connects**

1. Frontend sends a POST request with the ticker using Axios.
2. Backend (Django):

* Loads or trains the LSTM model.
* Predicts the stock price and calculates risk.

1. The response is sent back to the Frontend, where it can be displayed in the UI.