**Title: Building a Real-Time Cryptocurrency Trading Algorithm with LSTM and Streamlit Dashboard**

In this blog, I will walk you through how I spent a focused night building a real-time cryptocurrency trading prediction algorithm. This article is not just a technical guide but a personal learning journey for anyone interested in combining finance, machine learning, and web dashboards.

### 🔄 Project Goal:

To build a real-time trading algorithm that:

* Collects live market data (e.g., BTC/USDT)
* Analyzes it using technical indicators like RSI and MACD
* Predicts the next 4-hour candle’s closing price using an LSTM model
* Provides Buy/Sell/Hold signals
* Visualizes everything on a Streamlit-powered dashboard

### 🧰 Step 1: Defining the Problem

My aim was to predict the next price movement based on recent technical indicator patterns. This is a time series forecasting problem, suitable for models like LSTM.

### 📊 Step 2: Data Collection from Binance

Using the ccxt library, I collected 4-hour OHLCV (Open, High, Low, Close, Volume) data for BTC/USDT:

import ccxt  
exchange = ccxt.binance()  
df = exchange.fetch\_ohlcv('BTC/USDT', '4h', limit=150)

### 🌐 Step 3: Feature Engineering (RSI & MACD)

With the ta library, I calculated:

* RSI (Relative Strength Index)
* MACD (Moving Average Convergence Divergence)

These features were used alongside closing prices to feed into the ML model.

### 🧪 Step 4: LSTM Model Training

I trained an LSTM model using TensorFlow. The model took the last 10 data points (each containing close, RSI, MACD, and MACD signal) and predicted the next close price.

model = Sequential()  
model.add(LSTM(64, input\_shape=(10, 4)))  
model.add(Dense(1))  
model.compile(optimizer='adam', loss='mse')

The model was saved as your\_model.h5, and the scaler used to normalize inputs was saved as scaler.pkl.

### 🔹 Step 5: Building the Dashboard with Streamlit

Using streamlit, I built a dashboard that:

* Displays the latest BTC/USDT candlestick chart
* Shows RSI and MACD below the chart
* Loads the LSTM model and predicts the next 4h price
* Displays BUY / SELL / HOLD signals based on the predicted price

It also auto-refreshes every 5 minutes for live updates.

### 🚀 Bonus: TradingView Chart Integration

To make it look professional, I embedded a live TradingView chart using their public widget inside the Streamlit app.

st.components.v1.html(tradingview\_html\_code, height=620)

### 🌟 Key Technologies Used:

* Python
* CCXT (for Binance API)
* TA-Lib / ta (technical indicators)
* TensorFlow (LSTM model)
* Streamlit (dashboard)
* TradingView widget (real-time chart)

### 🚀 What I Learned:

* How to process real-time financial data
* Applying ML to time series forecasting
* Combining backend logic and frontend UI with Streamlit

### 📆 What’s Next?

* Add backtesting and performance metrics
* Deploy this dashboard online (Heroku, Render, or Streamlit Cloud)
* Expand to other coins like ETH, BNB, etc.

This project not only enhanced my understanding of financial markets and machine learning, but also prepared me for real-world AI applications in fintech. Feel free to connect and ask questions!

*#DataScience #MachineLearning #Crypto #LSTM #Streamlit #TradingAlgorithm #Binance #Python #FinTech*