This is the critical transition from **Hobbyist** to **Vendor**. You cannot sell a "Theory"; you can only sell a "Result."

Paper Trading is the process of generating that Result without risking capital. It creates the **Proof of Concept (PoC)** required to value your algorithm.

Here is the execution plan to turn your code into a sellable asset.

PHASE 1: THE PAPER TRADING SCRIPT (The Evidence Generator)

We need to modify your script. Instead of just *finding* a Buy Signal, it must now simulate the **Exit**.

The Logic:

1. **Entry:** When Score == 4 (Buy).
2. **Simulation:** Look forward in time (using historical data) or track live.
3. **Exit Rules:**
   - **Take Profit:** Sell if price rises **3%**.
   - **Stop Loss:** Sell if price drops **1.5%**.
4. **Result:** Did this signal make money?

The Script Update:
This script runs a "Backtest" on the signals found to prove if they would have made money.

Python

```
import yfinance as yf
import pandas as pd

def proof_of_concept_runner(ticker_list):
    print("--- GENERATING PROOF OF CONCEPT (BACKTEST) ---")

    ledger = []

    for symbol in ticker_list:
        try:
            # Get data
            df = yf.download(symbol, period="6mo", interval="1d", progress=False)
```

```
        if df.empty: continue

        # Calculate Indicators (Simplified for brevity)
        df['SMA_20'] = df['Close'].rolling(window=20).mean()
        df['Lower_Band'] = df['SMA_20'] - (df['Close'].rolling(window=20).std() * 2)

        # SIMULATION LOOP (Iterate through past days)
        # We skip the first 30 days to let indicators settle
        for i in range(30, len(df)-5):
            row = df.iloc[i]

            # THE BUY LOGIC (Simplified Triple Threat)
            # Price below band AND Volume spike
            if row['Close'] < row['Lower_Band'] and row['Volume'] > df['Volume'].iloc[i-1]:

                entry_price = row['Close']
                entry_date = df.index[i]

                # THE EXIT LOGIC (Look ahead 5 days)
                outcome = "HOLD"
                pnl = 0

                for j in range(1, 6): # Check next 5 days
                    future_price = df['Close'].iloc[i+j]
                    pct_change = (future_price - entry_price) / entry_price

                    if pct_change >= 0.03: # +3% Gain
                        outcome = "WIN"
                        pnl = 3.0
                        break
                    elif pct_change <= -0.015: # -1.5% Loss
                        outcome = "LOSS"
                        pnl = -1.5
                        break

                if outcome != "HOLD":
                    ledger.append({
                        "Ticker": symbol,
                        "Date": entry_date,
                        "Entry": round(entry_price, 2),
                        "Result": outcome,
                        "PnL_%": pnl
                    })
```

```
            except: continue

    # CALCULATE VALUE
    if ledger:
        results = pd.DataFrame(ledger)
        total_trades = len(results)
        wins = len(results[results['Result'] == 'WIN'])
        win_rate = (wins / total_trades) * 100

        print(f"\n--- ALGORITHM VALUATION METRICS ---")
        print(f"TOTAL TRADES SIMULATED: {total_trades}")
        print(f"WIN RATE: {round(win_rate, 2)}%")
        print(f"NET THEORETICAL GAIN: {results['PnL_%'].sum()}%")

        if win_rate > 60:
            print(">>> STATUS: MARKETABLE ASSET (High Value)")
        else:
            print(">>> STATUS: NEEDS REFINEMENT (Low Value)")

    else:
        print("No trades triggered in backtest.")

# EXECUTE
stock_list = ['TSLA', 'NVDA', 'AMD', 'COIN', 'MARA']
proof_of_concept_runner(stock_list)
```

PHASE 2: DEFINING THE VALUE OF THE ALGORITHM

When you sell an algorithm, you are selling **Certainty** and **Time**. The price is determined by these metrics:

Metric	Definition	Value Impact
Win Rate	Percentage of trades that are profitable.	**> 60%** = High Value. **> 75%** = Institutional Value.

Profit Factor	(Gross Profit) / (Gross Loss).	If you make $2 for every $1 lost, your factor is 2.0.
Drawdown	The maximum drop in account value.	Low Drawdown (<10%) increases value significantly.
The "Black Box" Factor	Is the logic unique?	Your "Triple Threat + VIX" logic is understandable but effective.

The Valuation Formula:
If your PoC model shows a monthly return of 10% on a $10,000 account:

- **Monthly Value:** $1,000.
- **Sale Price (Multiple):** Typically 12x to 24x monthly value.
- **Estimated Asset Price: $12,000 - $24,000**.

PHASE 3: BEST OPTIONS TO SELL THE ALGORITHM

You have four distinct paths to monetize this code.

1. The "Pickaxe" Model (SaaS/Subscription)

- **Concept:** You don't sell the code; you sell the **Signals**.
- **Method:** Set up a Telegram channel or Email list (using the script I gave you).
- **Price:** $29 - $49 / month per user.
- **Pros:** Recurring revenue. You keep the IP.
- **Cons:** Requires marketing.

2. The Marketplace Model (MQL5 / TradingView)

- **Concept:** Convert your logic into a plugin for popular trading platforms.

- **Method:** Translate your Python logic into **PineScript** (for TradingView) or **MQL** (for MetaTrader).
- **Price:** One-time fee of $50 - $200 per download.
- **Pros:** Access to millions of existing traders.
- **Cons:** High competition. Lower price point.

3. The Prop Firm Challenge (High Leverage)

- **Concept:** You don't sell the algorithm to others; you "sell" it to a Prop Firm (like FTMO or Apex).
- **Method:** You pay a fee to take a trading challenge. You use your Bot to pass the challenge. They give you a funded account (e.g., $100,000). You keep 80% of the profit.
- **Value:** If your bot makes 5% a month on $100k, you earn **$4,000/month**.
- **Pros:** Highest potential payout. No customers to manage.
- **Cons:** Strict rules (drawdown limits).

4. The White Label Sale (B2B)

- **Concept:** Sell the Python code directly to a small hedge fund or a "Guru" who wants to offer a bot to their community.
- **Method:** Networking on LinkedIn or specialized Discords.
- **Price:** $5,000 - $15,000 + Royalties.
- **Pros:** Large lump sum.
- **Cons:** You lose the rights to the code (usually).

COMPARISON TABLE

Model	Difficulty to Start	Potential Revenue	Scalability
SaaS (Signals)	Medium	Medium (Recurring)	High
Marketplace	Low	Low (Volume based)	Very High

Prop Firm	High (Strict Rules)	**Very High**	Low (Personal limit)
White Label	High (Sales skills)	High (Lump Sum)	Low

NEXT STEP

To proceed with the **Proof of Concept**, do you want to:

1. **Refine the Backtest:** Adjust the script to output a visual chart of the "Account Balance" over time (The Equity Curve)?
2. **Explore Prop Firms:** Get a list of firms that allow Python bots so you can attempt to win a funded account?