**Project Brief: Swing Trading Learning & Execution System (v1.1 - Final)**

## **1. Core Problem**

* Need a structured system and platform to effectively **learn** swing trading principles and practices.
* Desire a rules-based environment to enforce **trading discipline** and automate adherence to trading plans.
* Lack of integrated tools for systematic trade identification, planning, analysis (technical & fundamental), risk management, strategy development, and performance evaluation.
* Need a safe environment (paper trading) to practice and validate strategies before risking real capital.

## **2. High-Level Goals**

* Master a defined, systematic **process** for swing trading, covering identification, planning, execution, and post-trade analysis to find a personalized, effective niche.
* Achieve consistent **profitability** by systematically identifying, validating (via backtesting and quantitative analysis), and deploying strategies with statistically favorable expectancy (optimizing for probability of success and/or reward-to-risk ratio).
* Develop and maintain a personalized, locally-run platform that supports this entire trading **system** (rules, processes, strategies).

## **3. Target Audience**

* Primary User: Yourself - an individual with a background in Python, ML (Tensorflow certified), and extensive BA experience, seeking to learn and apply swing trading systematically.

## **4. Core Concept/Features (High-Level - Full Vision)**

* Learning Platform
* Trade Identification Module(s)
* Trading Plan Automation & Discipline Enforcement
* Research Integration (LLM for Fundamental Analysis)
* Advanced Risk Management Module (Algorithms, Volatility-Adjusted Sizing, Dynamic Adjustments)
* Strategy Development Environment (Backtesting, Optimization, Robust Validation - WFO/Time Series CV)
* Market Regime Detection Module
* Advanced Stock Screening Module
* Watchlist Management Module
* Visualization/Charting Tools
* Trading Simulation (Paper Trading) & Live Trading Capabilities
* Performance Evaluation (Enhanced Journaling with Psychology, Structured Post-Mortem Analysis)
* Support for Multiple Strategies (Breakouts, Pullbacks, EPs, Mean Reversion, etc.)

## **5. MVP Scope**

### **IN SCOPE for MVP:**

* **Data Handling:** Load **daily OHLCV** data for selected US stocks into **PostgreSQL/TimeScaleDB** database. Include basic **CSV import/export** functionality.
* **Visualization:** Basic interactive charting displaying Candlesticks, Volume, and selectable Moving Averages (e.g., 10, 20, 50-day), with indicators being togglable.
* **Strategy Implementation & Backtesting:** Ability to code and backtest the **"Classic Breakout"** (Continuation Base Breakout) strategy, using a selected Python backtesting framework (e.g., Backtrader). Output basic performance metrics (e.g., Total P/L, Win Rate, Max Drawdown).
* **Structured Trade Logging/Journaling:** Simple interface/module for manually logging trade ideas, plans (entry, stop, target), rationale, execution details, P/L, and **explicit fields for "Emotional State (Pre-Trade, During, Post-Trade)," "Mistakes Made," and "Lessons Learned"** to support psychological awareness and future analysis.
* **Basic Risk Feedback:** Within the logging/planning module, display the **calculated planned Risk per Trade (as % of capital)** and the **planned Risk:Reward ratio** based on the user's input entry, stop, and target levels.
* **Paper Trading Connection:** Basic connection to **Alpaca Markets** API to fetch live price data for observation. (Manual order submission based on system signals).
* **Trading Plan Checklist:** Simple **static checklist** (potentially via Django web UI) outlining predefined steps for trade consideration, with checklist data **stored in PostgreSQL**.
* **Educational Guidance (Basic):** Simple contextual links or references back to relevant uploaded research documents or key concept definitions related to the Classic Breakout strategy.

### **OUT OF SCOPE for MVP:**

* Advanced database schemas/queries beyond basic OHLCV storage and checklist data.
* Advanced charting features.
* Support for multiple simultaneous strategies or strategy optimization tools (e.g., walk-forward testing ).
* LLM Research Analyst integration.
* Advanced/automated risk management algorithms.
* Fully automated trade execution via API.
* Advanced performance analytics dashboards or dedicated post-mortem analysis modules.
* Live trading connectivity.
* Support for instruments beyond initial stock focus.
* Interactive educational modules/tutorials within the platform.
* Market Regime Detection Module implementation.
* Advanced Scanning Module implementation.
* Watchlist Management Module implementation.

## **6. Initial Technical Leanings**

* **Platform:** Local execution on MacBook M3 Pro.
* **Language/Environment:** Python, Anaconda.
* **Key Libraries:** Tensorflow, Pandas, NumPy, Matplotlib, Scikit-learn, TimescaleDB libraries, TA libraries (pandas-ta recommended initially ), Backtrader (or other selected backtesting lib), potentially Django (for checklist UI).
* **Database:** PostgreSQL with TimeScaleDB extension.
* **Development:** Notebooks for testing/dev.