# **Project Description - Factor-Based Stock Recommendation Tool with LLM Integration**

The pictures shows a simple backtesting tool I built using DuckDB, Streamit, pyplot for a simple **moving average crossover strategy** on a single stock. Now, we can generalize it into a **factor-based stock selection system**, which is much closer to how professional investment funds work.

### **System Design**

* **Data Layer**: Use DuckDB to store daily stock data (Open, High, Low, Close, Volume). Add tables for factor values such as momentum, volatility, and value. Think of each factor as a **function** that takes historical data as input and outputs a score for each stock.
* **Strategy Layer**: At each rebalance date (like a scheduled job), compute factor scores, rank all stocks, select the top N stocks, and assign portfolio weights (equal-weight, value-weighted, or factor-weighted).
* **Execution & Backtest Engine**: Reuse the existing backtesting logic (equity curve, trade charts, performance metrics). The difference is that now it runs at the **portfolio level** instead of just one stock.
* **Visualization Layer**: Keep current charts, but add new ones such as factor performance vs. benchmark, turnover heatmaps, or IC/IR statistics (basically evaluation metrics for factors).
* **UI**:
* By default, expose common factors like **Momentum**, **Volatility**, and **Value** with simple sliders and inputs of number stock selected, rebanlance frequency for basic users.
* Advanced users can either:

1. Write **custom formulas** for factors (like writing your own function).
2. Use an **LLM** to generate factor definitions or even full stock-selection strategies from natural language.

**Example with LLM**:

“At each rebalance, pick the top 20 momentum stocks but exclude those with PE > 30.”  
 The LLM generates SQL/Python, and the system runs it only after strict safety checks—IR validation, schema whitelisting, and read-only, parameterized execution—to ensure secure backtest results. This ensures the system remains safe from SQL injection, hallucinated fields, or other unsafe queries.

### **Features of LLM**

* **Factor Generation**: Convert plain language into executable formulas or queries.
* **External Data Fusion**: Incorporate signals from news or company reports as extra features (like sentiment scores).
* **Custom Training**: Fine-tune a smaller model on factor definitions, or use RAG so the model can always reference a factor library, with continuous evaluation and schema-aware prompts to reduce hallucinations and improve reliability.
* **Automation**: Move toward AutoML-style systems that automatically test and optimize factor combinations (advanced feature).

### **UI vs. LLM**

* **UI Control**: Sliders and inputs for a small number of basic factors.
* **Custom Code**: Users can write their own formulas if they want more control.
* **LLM Option**: Users describe strategies in plain English. The LLM compiles this into a structured pipeline, keeping the interface simple while still powerful.

### **Application of Linear Regression Model (Optional)**

Beyond manual or LLM-defined rules, we can also integrate a Linear Regression model into the system:

* Idea: Treat each factor (momentum, value, volatility, etc.) as a feature, and the stock’s future return as the target.
* Method: Run a regression at each rebalance date:
* Interpretation: The regression coefficients beta serve as weights that show how much each factor contributes to predicting returns.
* Portfolio Construction: Rank stocks by their regression-predicted returns, and build the porfolio accordingly.

This makes the tool not just a backtester, but also a machine learning application, where factor weights are learned from data instead fo set by hand

### **Real-World Connection**

The project workflow mirrors professional fund management:

* **Index Funds**: Use fixed rules (e.g., S&P 500 membership) and rebalance periodically.
* **Smart Beta / Factor ETFs**: Use rules based on factors (e.g., dividend yield, low volatility), validate with backtests, and rebalance monthly or quarterly.
* **Quant Hedge Funds**: Build proprietary multi-factor models, stress test them through backtests, and deploy in live trading.

**Key cycle**: define factors → validate with backtests → construct portfolios → rebalance.

### **Historical Context (Algorithm Analogy)**

Factor investing has evolved like computer algorithms:

* **1970s – “Hello World” stage**
* CAPM: only one factor (market beta).
* Analogy: like the very first sort() algorithm.
* **1990s – Multi-Factor Models**
* Fama-French 3-factor model (market, size, value).
* Analogy: like moving from bubble sort to quicksort — more practical and robust.
* **2000s – Institutional Adoption**
* Hedge funds build factor libraries (momentum, quality, etc.) with SAS/Matlab/R.
* Analogy: algorithms go from academic papers to production systems.
* **2010s – ETFs & Smart Beta**
* Factor ETFs (low volatility, high dividend, momentum).
* Analogy: pre-packaged libraries on PyPI — ready to install and use.
* **2020s – AI & Alternative Data**
* Use of news, NLP, satellite images, and LLMs for factor design.
* Analogy: moving from hand-coded algorithms to ML/AutoML/LLMs.

### **Key Takeaway**

Factor-based investing has been practiced for **50+ years**. Our project demonstrates a modern version:

* Default factors for beginners (momentum, volatility, value).
* Custom factors for advanced users (like writing your own function).
* Linear Regression to learn optimal factor weights from data.
* LLM integration for natural language → strategy automation.

In short, it’s a **mini quant research platform** that connects classic finance with modern machine learning and AI.

Reference:

<https://www.investopedia.com/terms/f/factor-investing.asp#toc-example-the-fama-french-3-factor-model>  
 <https://www.ssga.com/au/en_gb/intermediary/insights/education/factor-based-investing>

Understanding Factor Investing: A Strategy for Market Savvy Investors

Factor investing is an investing strategy that aims to manage risk and generate above-market returns by using multiple factors to analyze asset prices.