**Schema Lengkap Aplikasi TradingView Python: Dari Awal Sampai Akhir**

Berikut adalah **schema komprehensif end-to-end** untuk membangun aplikasi TradingView-style menggunakan Python 3.14 dengan fokus pada analisis trendline average multi-timeframe (top-down) dan deteksi support/resistance kuat. Schema ini mencakup seluruh tahapan dari perencanaan hingga deployment dan monitoring sistem trading yang akurat dan profesional.

**Ringkasan Eksekutif**

Schema aplikasi TradingView Python terdiri dari **15 tahapan utama** yang terstruktur dalam **4 layer arsitektur**: Presentation Layer (visualisasi chart), Business Logic Layer (analisis multi-timeframe dan indikator), Data Layer (akuisisi dan storage data), dan Infrastructure Layer (event-driven, monitoring, deployment). Setiap tahapan dirancang modular dan scalable untuk memudahkan maintenance serta pengembangan fitur lanjutan.[[1]](#fn1)[[2]](#fn2)[[3]](#fn3)[[4]](#fn4)

chart:71

**Tahapan Pengembangan Lengkap**

**Fase 1: Planning & Setup (Tahap 1-2)**

**1. Requirement & Planning**

Tahap awal adalah mendefinisikan requirement lengkap aplikasi. Tentukan simbol trading yang akan dimonitor (saham, crypto, forex), timeframe analisis (weekly, daily, 4H, 1H), indikator teknikal yang diperlukan (SMA/EMA, RSI, MACD, Fibonacci, Volume Profile, Pivot Points), serta fitur utama seperti analisis top-down multi-timeframe dan deteksi support/resistance kuat dengan strength rating. Dokumentasikan requirement dalam flowchart sistem dan spesifikasi teknis yang jelas.[[1]](#fn1)[[5]](#fn5)[[4]](#fn4)

**2. Environment Setup**

Install Python 3.14 dan setup virtual environment menggunakan virtualenv atau conda untuk isolasi dependency. Install library-library utama: **yfinance** atau **tvDatafeed** untuk data acquisition, **pandas** dan **numpy** untuk manipulasi data, **trendln** dan **pytrendline** untuk deteksi support/resistance, **lightweight-charts-python** untuk visualisasi TradingView-style, serta **pandas\_ta** atau **TA-Lib** untuk kalkulasi indikator teknikal. Pastikan semua library compatible dengan Python 3.14.[[6]](#fn6)[[7]](#fn7)[[8]](#fn8)[[9]](#fn9)

**Fase 2: Data Management (Tahap 3-4)**

**3. Data Acquisition Module**

Buat modul untuk fetch data historis OHLCV (Open, High, Low, Close, Volume) dari berbagai sumber API seperti Yahoo Finance (yfinance), Binance (ccxt), Alpha Vantage, atau TradingView API langsung. Modul ini harus mampu handle error, retry mechanism, dan rate limiting dari API. Implementasikan caching untuk mengurangi API calls yang berlebihan dan mempercepat load time.[[1]](#fn1)[[2]](#fn2)[[7]](#fn7)

**4. Data Processing & Storage**

Simpan data yang sudah di-fetch ke dalam struktur **pandas DataFrame** dengan index datetime dan kolom standar OHLCV. Untuk aplikasi real-time atau yang memerlukan persistence, pertimbangkan integrasi dengan database seperti SQLite untuk lightweight storage atau PostgreSQL untuk production-grade system. Implementasikan data validation untuk memastikan tidak ada missing values atau anomali data yang bisa mengganggu analisis.[[2]](#fn2)[[4]](#fn4)[[1]](#fn1)

code:70

**Fase 3: Analysis Engine (Tahap 5-8)**

**5. Technical Indicators Module**

Implementasikan kalkulasi berbagai indikator teknikal yang relevan. **SMA (Simple Moving Average)** untuk periode 20, 50, dan 200 sebagai trendline average utama. **EMA (Exponential Moving Average)** dengan periode Fibonacci (13, 21, 34, 55, 89) untuk dynamic support/resistance. **RSI** untuk identifikasi overbought/oversold. **MACD** untuk konfirmasi momentum. **Fibonacci Retracement** untuk level retracement potensial. **Volume Profile** dan **Pivot Points** untuk konfirmasi support/resistance. Gunakan library pandas\_ta atau TA-Lib untuk efisiensi komputasi.[[1]](#fn1)[[10]](#fn10)[[11]](#fn11)[[12]](#fn12)[[13]](#fn13)

**6. Multi-Timeframe Analysis Engine**

Implementasikan engine untuk resample data ke berbagai timeframe menggunakan fungsi **pandas resample**. Timeframe yang direkomendasikan: Weekly (1W) untuk trend global, Daily (1D) untuk trend medium-term, 4-Hour (4H) untuk swing trading, dan 1-Hour (1H) atau lebih rendah untuk intraday entry. Untuk setiap timeframe, hitung ulang semua indikator dan simpan dalam dictionary atau DataFrame terpisah. Engine ini menjadi fondasi analisis top-down yang memungkinkan validasi sinyal dari timeframe tertinggi hingga terendah.[[14]](#fn14)[[15]](#fn15)[[16]](#fn16)

**7. Support/Resistance Detection**

Modul krusial untuk identifikasi level support dan resistance yang kuat. Implementasikan beberapa metode deteksi:

* **Algoritma Wick Detection:** Deteksi candlestick dengan lower wick panjang (>2x body) sebagai support dan upper wick panjang sebagai resistance.[[10]](#fn10)[[13]](#fn13)[[17]](#fn17)
* **Pivot Points:** Kalkulasi pivot points klasik dan level S1, S2, R1, R2.[[11]](#fn11)[[18]](#fn18)
* **Clustering Algorithm:** Kelompokkan level-level yang berdekatan (threshold 2%) dan hitung **strength rating** berdasarkan frekuensi pengujian level tersebut.[[8]](#fn8)[[17]](#fn17)[[19]](#fn19)

Gunakan library **trendln** atau **pytrendline** sebagai baseline, kemudian customize dengan algoritma clustering untuk mendapatkan level terkuat.[[9]](#fn9)[[19]](#fn19)[[8]](#fn8)

**8. Trendline Average Calculation**

Hitung trendline average menggunakan SMA/EMA sebagai dynamic support/resistance. Trendline ini tidak hanya berfungsi sebagai indikator trend direction, tetapi juga sebagai level support saat uptrend (price di atas SMA) dan resistance saat downtrend (price di bawah SMA). Implementasikan juga linear regression trendline menggunakan **numpy polyfit** untuk identifikasi channel dan breakout potensial.[[12]](#fn12)[[16]](#fn16)[[20]](#fn20)[[21]](#fn21)[[8]](#fn8)

**Fase 4: Signal Generation & Visualization (Tahap 9-10)**

**9. Signal Generation Module**

Buat modul untuk generate sinyal trading berdasarkan konfirmasi multi-timeframe dan confluence dari berbagai indikator. Contoh logika sinyal buy: SMA20 > SMA50 (trend bullish), RSI < 70 (belum overbought), MACD bullish crossover, price mendekati support level kuat dari timeframe lebih tinggi, dan volume meningkat. Sinyal sell menggunakan kondisi kebalikan. Modul ini bisa rule-based untuk simplicity atau integrasikan machine learning model untuk adaptasi dinamis terhadap kondisi pasar.[[1]](#fn1)[[2]](#fn2)[[22]](#fn22)[[10]](#fn10)[[12]](#fn12)[[13]](#fn13)[[17]](#fn17)

**10. Chart Visualization (UI)**

Implementasikan visualisasi TradingView-style menggunakan **lightweight-charts-python**. Library ini menyediakan interface yang sangat mirip dengan TradingView dengan performa rendering yang baik. Tampilkan candlestick chart, overlay SMA/EMA lines dengan warna berbeda, plot level support/resistance sebagai horizontal lines dengan label strength rating, dan tambahkan marker untuk sinyal buy/sell. Chart harus interaktif dengan zoom, pan, dan timeframe switcher.[[6]](#fn6)[[23]](#fn23)[[7]](#fn7)[[24]](#fn24)

chart:74

**Fase 5: Real-time & Architecture (Tahap 11-13)**

**11. Real-time Data Handler**

Untuk aplikasi real-time, integrasikan **WebSocket** atau polling REST API untuk update data secara kontinyu. Gunakan **websocket-client** library dan **asyncio** untuk asynchronous handling agar tidak blocking main thread. Implementasikan callback functions yang akan trigger update chart dan recalculate indicators setiap ada data baru. Pertimbangkan throttling untuk menghindari overload computational saat market volatil.[[2]](#fn2)[[4]](#fn4)[[23]](#fn23)

**12. Event-Driven Architecture**

Implementasikan event-driven system sebagai backbone aplikasi. Setiap aksi penting (data baru, sinyal trading, perubahan position) generate event yang di-handle oleh event handlers spesifik. Gunakan **Python Queue** dan **threading** untuk event processing loop. Arsitektur ini membuat sistem modular, mudah debugging, dan scalable karena setiap komponen berkomunikasi via events tanpa tight coupling. Contoh event: BalanceUpdate, NewDataReceived, SignalGenerated, OrderFilled.[[4]](#fn4)[[25]](#fn25)[[2]](#fn2)

**13. Risk Management Module**

Modul kritis untuk proteksi kapital. Implementasikan: **position sizing** berdasarkan risk percentage (contoh: risk 1-2% per trade), **stop-loss** otomatis pada level support/resistance terdekat atau berdasarkan ATR, **take-profit** pada resistance level berikutnya atau risk-reward ratio tertentu (misal 1:2 atau 1:3), **max drawdown limit** untuk stop trading saat kerugian kumulatif mencapai threshold, **leverage control**, dan **max position per symbol**. Module ini harus terintegrasi dengan signal generator untuk validasi setiap sinyal sebelum eksekusi.[[5]](#fn5)[[2]](#fn2)[[4]](#fn4)

**Fase 6: Testing & Deployment (Tahap 14-15)**

**14. Testing & Quality Assurance**

Lakukan testing komprehensif di berbagai level. **Unit testing** dengan **pytest** untuk setiap modul (indicators calculation, support/resistance detection, signal logic). **Integration testing** untuk validasi API connections dan data flow. **Backtesting** menggunakan library [**backtesting.py**](http://backtesting.py) untuk test strategi dengan data historis dan evaluasi metrics seperti Sharpe Ratio, win rate, max drawdown. **Security audit** untuk proteksi API keys dengan environment variables. Pastikan semua edge cases ter-handle dengan baik.[[1]](#fn1)[[2]](#fn2)[[15]](#fn15)

**15. Deployment & Monitoring**

Deploy aplikasi ke server cloud (AWS EC2, GCP Compute Engine, atau VPS) menggunakan **Docker** untuk containerization dan reproducibility. Setup **systemd** atau supervisor untuk automated restart on failure. Implementasikan comprehensive logging dengan **loguru** dan monitoring dengan tools seperti **Prometheus** atau custom dashboard. Setup alerts untuk critical errors, abnormal trading patterns, atau API failures. Pastikan aplikasi bisa run 24/7 dengan minimal downtime.[[2]](#fn2)[[5]](#fn5)[[4]](#fn4)[[1]](#fn1)

**Arsitektur Sistem 4-Layer**

chart:74

Arsitektur sistem menggunakan **layered architecture pattern** untuk separation of concerns yang jelas:

1. **Presentation Layer:** Chart visualization dengan lightweight-charts, user interface untuk parameter input, dan event markers untuk sinyal trading.[[3]](#fn3)[[4]](#fn4)
2. **Business Logic Layer:** Core analysis engine yang berisi multi-timeframe analyzer, technical indicators calculator, support/resistance detector, signal generator, dan risk manager.[[1]](#fn1)[[4]](#fn4)[[3]](#fn3)
3. **Data Layer:** Data acquisition module untuk fetch data dari API, data storage dengan pandas/database, dan real-time data handler untuk streaming updates.[[2]](#fn2)[[4]](#fn4)[[1]](#fn1)[[3]](#fn3)
4. **Infrastructure Layer:** Event-driven architecture untuk orchestration, logging & monitoring untuk observability, dan deployment infrastructure untuk production readiness.[[4]](#fn4)[[25]](#fn25)[[2]](#fn2)[[3]](#fn3)

code:73

**Contoh Implementasi Kode Lengkap**

code:72

Kode aplikasi lengkap di atas mendemonstrasikan implementasi modular dengan **8 komponen utama**:

1. **DataModule:** Fetch data OHLCV dari Yahoo Finance dengan error handling
2. **IndicatorsModule:** Kalkulasi SMA, EMA, RSI, MACD, Pivot Points dengan pandas\_ta
3. **MultiTimeframeEngine:** Resample data ke Weekly, Daily, 4H dengan pandas resample
4. **SupportResistanceDetector:** Deteksi level S/R dengan wick detection dan clustering algorithm
5. **SignalGenerator:** Generate sinyal buy/sell dengan multi-indicator confirmation
6. **ChartVisualizer:** Visualisasi TradingView-style dengan lightweight-charts-python
7. **TradingViewApp:** Main application class yang orchestrate seluruh workflow
8. **Main Execution:** Entry point untuk run aplikasi dengan parameter customizable

Total implementasi sekitar **250+ lines** kode Python yang clean, modular, dan mudah di-extend.[[1]](#fn1)[[2]](#fn2)[[4]](#fn4)

**Best Practices & Recommendations**

**Pilihan Indikator untuk Konfirmasi Support/Resistance Kuat:**

* **Volume Profile:** Konfirmasi level dengan highest trading volume (Point of Control).[[12]](#fn12)[[26]](#fn26)
* **Fibonacci Retracement:** Level 38.2%, 50%, 61.8% sering bertindak sebagai S/R kuat.[[11]](#fn11)[[12]](#fn12)
* **Moving Averages:** SMA 50/200 sebagai dynamic S/R terutama di trend kuat.[[16]](#fn16)[[12]](#fn12)
* **RSI Divergence:** Konfirmasi reversal di level S/R dengan RSI divergence.[[10]](#fn10)[[13]](#fn13)
* **Pivot Points:** S1/S2/R1/R2 sebagai intraday S/R reference.[[18]](#fn18)[[11]](#fn11)

**Tips Analisis Top-Down Multi-Timeframe:**

1. Mulai dari timeframe tertinggi (Weekly) untuk identifikasi major trend dan level S/R utama
2. Turun ke timeframe medium (Daily) untuk konfirmasi dan refinement level
3. Gunakan timeframe terendah (4H/1H) hanya untuk timing entry yang presisi
4. Hanya ambil posisi saat semua timeframe aligned (confluence)[[14]](#fn14)[[17]](#fn17)[[16]](#fn16)

**Optimasi Performance:**

* Cache data historis untuk mengurangi API calls
* Gunakan vectorized operations pandas/numpy untuk speed
* Implementasi async processing untuk real-time data
* Database indexing jika menggunakan persistent storage[[1]](#fn1)[[2]](#fn2)[[4]](#fn4)

**Kesimpulan**

Schema lengkap di atas menyediakan **roadmap komprehensif** untuk membangun aplikasi TradingView Python yang simple namun detail dan akurat. Dengan mengikuti 15 tahapan terstruktur, mengimplementasikan arsitektur 4-layer yang solid, dan memanfaatkan library-library terbaik untuk setiap komponen, Anda bisa membangun sistem trading yang powerful untuk analisis multi-timeframe dengan deteksi support/resistance yang akurat.

Aplikasi hasil implementasi schema ini akan mampu melakukan analisis top-down dari timeframe tertinggi hingga terendah, mengidentifikasi level support/resistance terkuat dengan strength rating, menampilkan trendline average sebagai dynamic S/R, dan generate sinyal trading dengan konfirmasi multi-indikator—semua divisualisasikan dalam chart TradingView-style yang profesional dan interaktif.

⁂

1. <https://www.geeksforgeeks.org/python/automated-trading-using-python/>

1. <https://github.com/ThePredictiveDev/Automated-Financial-Market-Trading-System>

1. <http://www.turingfinance.com/algorithmic-trading-system-architecture-post/>

1. <https://dev.to/jungle_sven/simple-yet-effective-architecture-patterns-for-algorithmic-trading-5745>

1. <https://successive.tech/blog/stock-trading-app-step-by-step-guide/>

1. <https://github.com/filipemarques87/tradingview-chart-py>

1. <https://www.insightbig.com/post/replicating-tradingview-chart-in-python>

1. <https://github.com/GregoryMorse/trendln>

1. <https://github.com/ednunezg/pytrendline>

1. <https://blog.stackademic.com/support-resistance-and-rsi-automated-detection-in-python-36cac7a812e8>

1. <https://www.dominionmarkets.com/5-best-support-and-resistance-indicators/>

1. <https://www.luxalgo.com/blog/5-indicators-for-confirming-support-and-resistance/>

1. <https://wire.insiderfinance.io/python-trading-tutorial-how-to-use-rsi-with-support-and-resistance-for-accurate-signals-46a188e44c55>

1. <https://tradeciety.com/how-to-perform-a-multiple-time-frame-analysis>

1. [https://kernc.github.io/backtesting.py/doc/examples/Multiple Time Frames.html](https://kernc.github.io/backtesting.py/doc/examples/Multiple%20Time%20Frames.html)

1. <https://id.tradingview.com/script/UQt09gTD-Multi-timeframe-trend/>

1. <https://id.tradingview.com/script/joTGw5wM-Multi-TimeFrame-Support-and-Resistance-w-Strength-Rating/>

1. <https://www.mindmathmoney.com/articles/the-5-best-tradingview-support-amp-resistance-tools-for-accurate-technical-analysis-2025>

1. <https://github.com/BatuhanUsluel/Algorithmic-Support-and-Resistance>

1. <https://id.tradingview.com/scripts/trenddirection/>

1. <https://www.youtube.com/watch?v=wbFoefnidTU>

1. <https://towardsdatascience.com/how-to-create-a-fully-automated-ai-based-trading-system-with-python-708503c1a907/>

1. <https://www.youtube.com/watch?v=TlhDI3PforA>

1. <https://www.youtube.com/watch?v=u8SmESEtQqo>

1. <https://wire.insiderfinance.io/engineering-an-algotrading-framework-1-requirements-and-workflow-32877183304a>

1. <https://id.tradingview.com/scripts/supportandresistance/page-3/>

1. <https://python.plainenglish.io/depth-tutorial-the-fully-automated-python-backend-of-my-stock-app-4d5087b67001>

1. <https://www.youtube.com/watch?v=9Y3yaoi9rUQ>

1. <https://www.youtube.com/watch?v=O3O1z5hTdUM>

1. <https://github.com/stefan-jansen/machine-learning-for-trading>

1. <https://gaper.io/algorithmic-trading-in-python/>

1. <https://www.youtube.com/watch?v=LLhb2cMcgZ4>

1. <https://www.youtube.com/watch?v=lpaq_6kU5hg>

1. <https://www.tradingview.com/charting-library-docs/latest/ui_elements/drawings/drawings-api/>

1. <https://id.tradingview.com/scripts/python/>

1. <https://www.tradingview.com/pine-script-docs/faq/data-structures/>