

Walkthrough - Service Stabilization & Script Upgrade

The system services were unstable, with recurrent crashes in the API Gateway and failure to start for Prediction/Signal Processing services due to missing dependencies.

Recent Updates

1. Port Stabilization & Connectivity

- **Centralized Config:** Created `frontend/src/config.js` to manage all API and WebSocket URLs in one place.
- **Backend Alignment:** Updated `pipeline_runner.py` to correctly use the API Gateway port (8010).
- **Service Stack:** Restarted all services. The Frontend now strictly communicates via `http://localhost:8010/api/v1`.

2. Intraday Trading Refinement

- **Logic Update:** Adjusted `LevelCalculator` to use tighter multipliers fitting for Intraday trading.
 - **Stop Loss:** 0.25x ATR (~0.5% risk)
 - **Target 1:** 0.5x ATR (~1% reward)
 - **Target 2:** 0.8x ATR (~1.6% reward)
- **Rationale:** Previous calculation used a hardcoded 2% "Daily ATR" proxy, making targets too wide. New multipliers correct for this.

3. Frontend Fixes

- **Portfolio.jsx:** Resolved persistent failing syntax errors ("Missing semicolon", "Unexpected character ₹") by:
 - Replacing template literal header construction with explicit `if/else` block.
 - Fixing Rupee symbol encoding inside template literals.
 - Removing accidental markdown delimiters inserted during editing.
- **Verification:** Confirmed successful page load in browser (`localhost:3000/portfolio`) with live data visibility.

4. Dashboard & Connectivity Fixes

- **Shadowing Bug:** Fixed `TypeError` in `Dashboard.jsx` where a local `config` object was overriding the global API configuration.
- **UI Polish:** Fixed "ANALYZING undefined" text in the progress bar by adding null-safety checks for the `symbol` property.
- **Gateway Stability:** Restarted API Gateway and verified it is listening on port 8010.
- **Verification:** Used browser subagent to confirm Dashboard loads, connects via WebSockets, and displays live signals without console errors.



Dashboard Verification

5. Dependency Fixes

- **Installed:** `xgboost` (for Prediction Service) and `textblob`, `nltk` (for Signal Processing Service).

5. Service Management

- **Created:** `manage_services.sh`
- **Capabilities:**
 - **Granular Control:** `./manage_services.sh [start|stop|restart] [service_name]`
 - **Full Stack:** `./manage_services.sh restart all`
 - **Docker Aware:** Checks for Docker presence before attempting container operations.

3. Usage Examples

```
# Restart everything
./manage_services.sh restart all
```

7. Global Market Sentiment (NEW)

- **Filter:** Dashboard now only displays High Conviction signals (> 60%).
- **Rationale:** AI now automatically includes insights from `Screener.in` (Fundamentals) and `TickerTape` (Forecast/Ratings) in the explanation text.
- **Scoring:** Integrated **Analyst Score (15% weight)** into the main model, boosting stocks with "Buy" ratings and high upside forecasts.
- **Rationale Improvements:**
 - **Unified View:** Consolidated Screener, TickerTape, and Fundamentals into a single "Fundamental Analysis" section.
 - **Color Coding:** Added semantic coloring (Green/Red/Orange/Cyan) for key sentiments, metrics (Oversold/Overbought), and analyst ratings.
 - **Stock Signal Color Coding:** Score breakdown labels (S, T, A, F, A) are now color-coded and bold.
- **AI Rationale Styling:** Keyword highlights (oversold, bullish, etc.) are now bright and bold across all themes.
- **Live News & Market Ticker:** A scrolling bar featuring live indices (Nifty, Sensex) and market news from yfinance (clickable) and LiveMint (non-clickable).

8. Intraday Level Refinement

- **Problem:** Some stocks (e.g., Maruti) were showing wide "Swing" targets (~8%) even after the logic update.
- **Root Cause:** Old recommendations were cached, and `lifecycle.py` was suppressing updates if AI confidence didn't strictly increase.
- **Fix:**
 - Updated `lifecycle.py` to allow updates when conviction is equal (`>=`).
 - Cleared `data/recommendations.json` to force a refresh.
- **Verification:** Verified via Dashboard scan. Maruti now shows a tight target of ₹14,796.71 (~1.3% distance).



Maruti Tight Targets

- **Data Source:** Real-time US/India Futures via `yfinance` (`ES=F` , `NQ=F` , `^NSEI`).
- **Scoring:**
 - **Bearish:** Global Futures < -0.3% → **-20% Penalty.**
 - **Bullish:** Global Futures > 0.3% → **+10% Boost.**

- **Currency Risk:** USD/INR > 0.2% (Rupee Weakening) → **-10% Penalty**.
- **ADR Overlay:** Strong ADR (> +1.5%) → **+5% Boost**; Weak ADR (< -1.5%) → **-5% Penalty**.
- **Rationale:** Added **Global Outlook** (e.g., "Global: Bearish | ADR (INFY): +1.2% (Bullish)") to the rationale text.

9. Score Details Visibility (NEW)

- **Problem:** Users could only see the total conviction percentage, with no transparency into how each component (Sentiment, Technical, AI Model, Fundamental, Analyst) contributed.
- **Solution:**
 - Expanded the backend `score_breakdown` in `main.py` to include all five core components.
 - Added mini-badges (S, T, A, F, AN) to each stock card in the Dashboard for immediate sub-score visibility.
 - Added a detailed "Sub-Score Breakdown" section to the strategy report modal for full transparency.
- **Verification:** Triggered a live scan and verified the presence of the new UI components. Applied a rounding fix to the modal values for a cleaner presentation.

```
![Stock Card with Mini-Badges]
(/Users/arunachalam/.gemini/antigravity/brain/6b10e4f9-1dfb-4c82-af40-9f04d8a9008f/card_badges_view_1769789161404.png)
<!-- slide -->
![Detailed Sub-Score Breakdown Modal]
(/Users/arunachalam/.gemini/antigravity/brain/6b10e4f9-1dfb-4c82-af40-9f04d8a9008f/modal_breakdown_view_1769789182405.png)
```

10. Configuration Centralization & Environment Portability (NEW)

To improve maintainability and ease of deployment, I refactored the entire system to use a centralized configuration architecture.

- **Centralized Config:** Expanded `shared/config.py` using Pydantic Settings to serve as the single source of truth for all ports, URLs, scoring weights, and thresholds.
- **Environment Management:** Created a template `.env` file for managing environment-specific overrides (ports, secrets, thresholds).
- **Service Refactoring:** Updated all core services (`recommendation_engine` , `api_gateway` , `ingestion_service` , `prediction_service`) to pull configurations dynamically from the centralized singleton.
- **Verification:** Confirmed services correctly load settings from `.env` and `shared/config.py` , and verified end-to-end connectivity.

11. USDINR Sentiment & Stock Refinement (NEW)

- **USDINR Analysis:** Verified that `GlobalMarketCrawler` fetches USD/INR data. A weakening Rupee now triggers a sentiment penalty for the broader market.
- **Rationale Visibility:** Improved `main.py` to explicitly display Rupee status (e.g., "Rupee: Strengthening (-0.25%)") in the AI rationale on the Dashboard.
- **Data Refresh:** Updated `recommendations.json` with 50+ new high-quality stock entries, covering major Nifty 50 constituents.

- **Manual Verification:** Confirmed that new recommendations correctly reflect the updated rationale format including fundamental insights.

12. India VIX & Volatility Integration (NEW)

- **Volatility Tracking:** Added `^INDIAVIX` to the `GlobalMarketCrawler`. The system now fetches absolute VIX values alongside traditional price change data.
- **Regime Intelligence:** Updated `ScoringModel` to automatically flag a `VOLATILE` market regime if India VIX exceeds 20.0.
- **Dynamic Level Scaling:** Refactored `LevelCalculator` to widened targets and stop-losses by **20% to 50%** when volatility is high, preventing premature stop-outs.
- **Risk Mitigation:** Integrated a VIX-based risk penalty (up to 40%) that reduces conviction scores during extreme market turbulence.
- **Visual Feedback:** Added a "**Volatility Alert**" to the AI rationale on the Dashboard, informing users when levels have been adjusted for VIX.

13. UI Aesthetics & Experience Refinement (NEW)

- **Color-Coded Badges:** The `S`, `T`, `A`, `F`, `A` (Sentiment, Technical, AI Model, Fundamental, Analyst) badges in stock cards are now color-coded and bolded for instant visual identification.
- **Enhanced Rationale View:** Refined the AI Strategy Report with brighter accents, bold cyan headers, and improved contrast to make complex insights easier to scan and read.
- **High-Contrast Styling:** Key market conditions and scorecard ratings are now visually highlighted to improve information hierarchy.
- **Theme-Aware Rationale:** Fixed an issue where white text was invisible in light mode by making the rationale text theme-responsive.

UI Progress Preview

```
![Stock Card Badges](/Users/arunachalam/.gemini/antigravity/brain/6b10e4f9-1dfb-4c82-af40-9f04d8a9008f/stock_card_badges_1769793326488.png)
<!-- slide -->
![AI Rationale View (Dark)](/Users/arunachalam/.gemini/antigravity/brain/6b10e4f9-1dfb-4c82-af40-9f04d8a9008f/ai_rationale_view_1769793342031.png)
<!-- slide -->
![AI Rationale View (Light)](/Users/arunachalam/.gemini/antigravity/brain/6b10e4f9-1dfb-4c82-af40-9f04d8a9008f/light_mode_dashboard_1769793627258.png)
```

Live News and Market Ticker

Added a scrolling ticker bar that provides real-time market indices and clickable news headlines. I also integrated LiveMint market news as requested.

 Ticker Bar Verification

- **Live Indices:** Nifty 50, Sensex, Bank Nifty, and India VIX with price movements.
- **Color-Coded Movements:** Positive changes are green, negative are red, with up/down arrows.
- **LiveMint Integration:** Fetches news from `livemint.com/market` with a non-clickable implementation.
- **Scrolling Animation:** Smooth horizontal scroll with pause-on-hover functionality.
- **Theme Support:** Consistent visibility in both light and dark modes.

Technical Specification

Created a comprehensive [technical_specification.md](#) providing in-depth details of the system's HLD and LLD.

- **Architecture Diagrams:** Mermaid based visualization of the microservices interaction.
- **Scoring Engine Deep-Dive:** Documentation of the S.T.A.F.A logic and weight distribution.
- **End-to-End Flows:** Detailed breakdown of the automated scanning and data processing pipeline.
- **Deployment Strategy:** Strategy for local (bash-based) and professional (containerized) environments.
- **Regime Detection:** Explanation of how the system adapts as the market regime changes (Chop/Trending/Volatile).