Of course. This is the final and most complete version of the prompt, incorporating the specific monetization model and embracing a modern, containerized deployment strategy with Docker.

This blueprint provides a comprehensive plan for building a commercially viable, scalable, and technically robust platform.

Definitive Project Prompt: ShareWise AI

Project Name: ShareWise AI - Multi-Broker, Explainable AI-Powered Trading Platform

I. Objective:

To build a fully automated and intelligent stock trading platform that **integrates with multiple Indian stock brokers** to:

1. Analyze the Indian share market using financial data, chart patterns, and news sentiment.
2. Recommend trading strategies and allow users to **build, train, and lease their own custom AI models**.
3. Execute trades automatically on user approval through their chosen broker.
4. Ensure risk-managed trading by focusing on **high-probability setups, statistical edge, and superior risk-adjusted returns**.
5. Create a sustainable commercial product based on a clear subscription and marketplace model.

II. Tech Stack:

- **Frontend:** React (for both web and PWA mobile wrapper)
- **Backend:** Django + Django REST Framework (DRF)
- **Database:** PostgreSQL
- **Object Storage:** AWS S3 (or similar, for storing trained model files)
- **Realtime Communication:** WebSockets / Django Channels
- **Broker APIs:** A **Broker Abstraction Layer** to support multiple brokers (e.g., Zerodha Kite Connect, AngelOne SmartAPI, Upstox API).
- **Payment Gateway:** Stripe or Razorpay integration.

- **ML Engine:** Python (pandas, scikit-learn, XGBoost, Prophet, TA-Lib, **SHAP**)
- **Deployment & Orchestration: Docker & Docker Compose**
- **Web Server / Reverse Proxy:** Nginx
- **UI/UX Theme:** Clean and professional design with a blue and white color palette

---

III. Core Features:

1. User Onboarding & Broker Authorization

- Signup/Login with email or mobile.
- **Connect & Authorize Broker Account:** Allow users to connect multiple accounts from a list of supported brokers.
- Securely store all encrypted API keys and access tokens.

2. Market Analysis Engine 🧠

- Fetches and analyzes market data.
- Generates signals from pre-built strategies using a **hybrid rule-based/ML model**.
- Signal Outputs include Confidence Score, Risk:Reward ratio, and an **Explainable AI (XAI) Justification**.

3. The AI Model Studio & Strategy Marketplace 🚀

- **AI Model Studio:** An intuitive, step-by-step interface for users to train their own custom ML models on the platform's data.
    - **Asynchronous Training:** Model training runs as a background job on dedicated Celery workers.
    - **Performance Dashboard:** A results page shows backtest P&L, Sharpe Ratio, and a Feature Importance chart (via SHAP).
- **Strategy Marketplace:**
    - Users can publish their validated and backtested models to a marketplace.
    - Other users can "lease" these models on a monthly basis.

4. Trading Automation

- **Broker Selection for Trades:** Users can assign strategies to specific broker accounts.
- Trade approval flow: Notify user → Confirm → Execute.
- Automated trade management (SL, Target, Time-based exit).
- Aggregated portfolio tracking across all connected brokers.

IV. Monetization Model 💰

The platform will operate on a Freemium SaaS model with a marketplace commission.

1. Subscription Tiers:

- **Free Tier:**
  - Limited to 5 backtests per day.
  - 1 live trading strategy active at a time.
  - No access to the AI Model Studio.
- **Pro Tier (₹1,000 / month):**
  - Up to 100 backtests per day.
  - Up to 10 live trading strategies.
  - Full access to the AI Model Studio to build and train models.
- **Elite Tier (₹2,500 / month):**
  - Unlimited backtests & live strategies.
  - Access to advanced features (e.g., institutional-grade pre-built models).
  - Ability to publish models to the Strategy Marketplace.

2. Marketplace Commission:

- A **10% commission** will be charged by the platform on all revenue generated by users leasing out their models in the marketplace.

V. Admin Panel (Django Admin)

- Manage users, global configurations, and pre-built strategies.

- **Manage Subscriptions & Payments:** View user tiers, payment history, and manage subscription statuses.
- **Oversee Marketplace:** Moderate submitted models and manage payouts to model creators.
- Monitor the status of all user-submitted training jobs.

VI. Reporting 📊

- Generate consolidated or per-broker reports.
- Track P&L, win rate, **Sharpe Ratio, Sortino Ratio, and Maximum Drawdown**.

VII. Compliance & Security

- Maintain a complete audit trail for all trades and user approvals.
- Implement top-tier security for user data, API keys, and payment information.
- Display clear disclaimers on risk and the probabilistic nature of signals.

VIII. Hosting & Deployment

- **Containerized Environment:** The entire application (Django, React, Celery, Nginx) will be containerized using **Docker**.
- **Orchestration: Docker Compose** will be used to manage the multi-container application in development and production.
- **Cloud Provider:** Hosted on a cloud platform like AWS or DigitalOcean.

IX. Developer Instructions:

1. **Containerize Everything:** Create Dockerfiles for each service (Django, React, Celery) and a docker-compose.yml file to orchestrate them.
2. **Build the Broker Abstraction Layer:** Design and build a unified interface to handle authentication and order execution for multiple brokers, starting with Zerodha.
3. **Implement Subscription & Payments:** Integrate a payment gateway (Stripe/Razorpay) to manage the Free, Pro, and Elite subscription tiers.

4. **Develop the AI Model Studio:** Build the intuitive UI and the asynchronous backend training pipeline using Celery workers.
5. **Create the Strategy Marketplace:** Develop the logic for users to publish, lease, and earn from their models, including the 10% commission system.
6. **Implement Explainable AI (XAI):** Use SHAP or a similar library to generate and display justifications for all AI-driven signals.
7. **Ensure Robust Security:** Apply best practices for data encryption, secure API design, and protecting sensitive credentials.
8. **Provide Comprehensive Documentation:** Document the architecture, API endpoints, and setup process, including all Docker configurations.