Stock Prediction System Software Specification

1. System Overview

1.1 Purpose

A distributed system for stock market prediction using multiple AI/ML models, orchestrated through a central traffic control agent.

1.2 High-Level Architecture

2. Component Specifications

2.1 Frontend (React)

- Single-page application
- Real-time data visualization
- User authentication and session management
- Responsive design for multiple device types
- State management (Redux/Context API)

2.2 Backend Service

- Flask-based REST API
- Endpoints:
    - `/api/v1/predict` - Get stock predictions
    - `/api/v1/health` - Service health check
    - `/api/v1/models` - Available models info
    - `/api/v1/history` - Historical predictions
- JWT authentication
- Request rate limiting
- Error handling and logging

2.3 Traffic Control Agent (TCA)

- gRPC server implementation
- Responsibilities:
    - Load balancing between models
    - Request routing
    - Model health monitoring
    - Result aggregation
    - Caching layer for frequent requests

- Configurable routing strategies
- Metrics collection

2.4 ML Agents

2.4.1 Common Agent Interface

All agents must implement:

- Prediction endpoint
- Health check
- Model metadata
- Performance metrics
- Error handling

2.4.2 Specific Agents

Hamiltonian Neural Network Agent

- Purpose: Capture conservation laws in price movements
- Input: Time series data
- Output: Price prediction with confidence interval
- Training frequency: Daily
- Metrics tracking:
    - MSE
    - Directional accuracy
    - Sharpe ratio

Fourier Neural Network Agent

- Purpose: Capture cyclical patterns
- Input: Time series data with technical indicators
- Output: Price prediction with frequency components
- Training frequency: Weekly
- Metrics tracking:
    - Frequency prediction accuracy
    - Phase alignment accuracy

Perturbation Theory Neural Network

- Purpose: Capture market regime changes
- Input: Time series data with market sentiment
- Output: Regime classification and price prediction
- Training frequency: Daily
- Metrics tracking:
    - Regime classification accuracy

    - Transition prediction accuracy

Generative LLM Agent

- Purpose: Incorporate news and sentiment analysis
- Input: Market news, social media sentiment, time series data
- Output: Market sentiment score and price impact prediction
- Update frequency: Real-time
- Metrics tracking:
    - Sentiment accuracy
    - News impact correlation

3. Data Flow

3.1 Prediction Request Flow

1. User submits prediction request through frontend
2. Backend validates request and forwards to TCA
3. TCA determines appropriate model(s)
4. Models generate predictions
5. TCA aggregates results
6. Response returned through chain

3.2 Data Requirements

- Historical price data (1-minute, 5-minute, daily)
- Trading volume
- Market sentiment data
- News feeds
- Technical indicators

4. Deployment Architecture

4.1 Container Strategy

- Each component in separate container
- Docker Compose for development
- Kubernetes for production
- Separate configurations for dev/staging/prod

4.2 Scaling Strategy

- Horizontal scaling for Frontend and Backend
- Model replication based on demand
- Cache layer for frequent requests

5. Testing Strategy

5.1 Unit Tests

- Each component requires 85%+ coverage
- Mock external dependencies
- Test configuration variations

5.2 Integration Tests

- End-to-end test scenarios
- Performance testing
- Load testing
- Fault tolerance testing

5.3 Model Testing

- Backtesting framework
- A/B testing capability
- Model comparison metrics
- Out-of-sample validation

6. Monitoring and Logging

6.1 Metrics

- System health metrics
- Model performance metrics
- Prediction accuracy metrics
- Resource utilization
- Request latency

6.2 Logging

- Centralized logging system
- Log levels: DEBUG, INFO, WARNING, ERROR
- Request/Response logging
- Model prediction logging
- Error tracking

7. Security Considerations

7.1 Authentication & Authorization

- JWT-based authentication
- Role-based access control
- API key management
- Rate limiting

7.2 Data Security

- Encryption at rest
- Encryption in transit
- Regular security audits
- Compliance with financial data regulations

8. Future Considerations

8.1 Potential Enhancements

- Additional model types
- Real-time trading integration
- Advanced ensemble methods
- Automated model selection
- Risk management module

8.2 Scale Considerations

- Multi-region deployment
- Data partitioning strategy
- Caching strategy
- Load balancing improvements