Fox ML Infrastructure [UNICODE] System Architecture Diagram

This document provides a visual and textual representation of the Fox ML Infrastructure system architecture.
This architecture diagram is essential for enterprise technical reviews and integration planning.

---

1. Executive Summary

Fox ML Infrastructure is a client-hosted, modular ML research infrastructure platform.

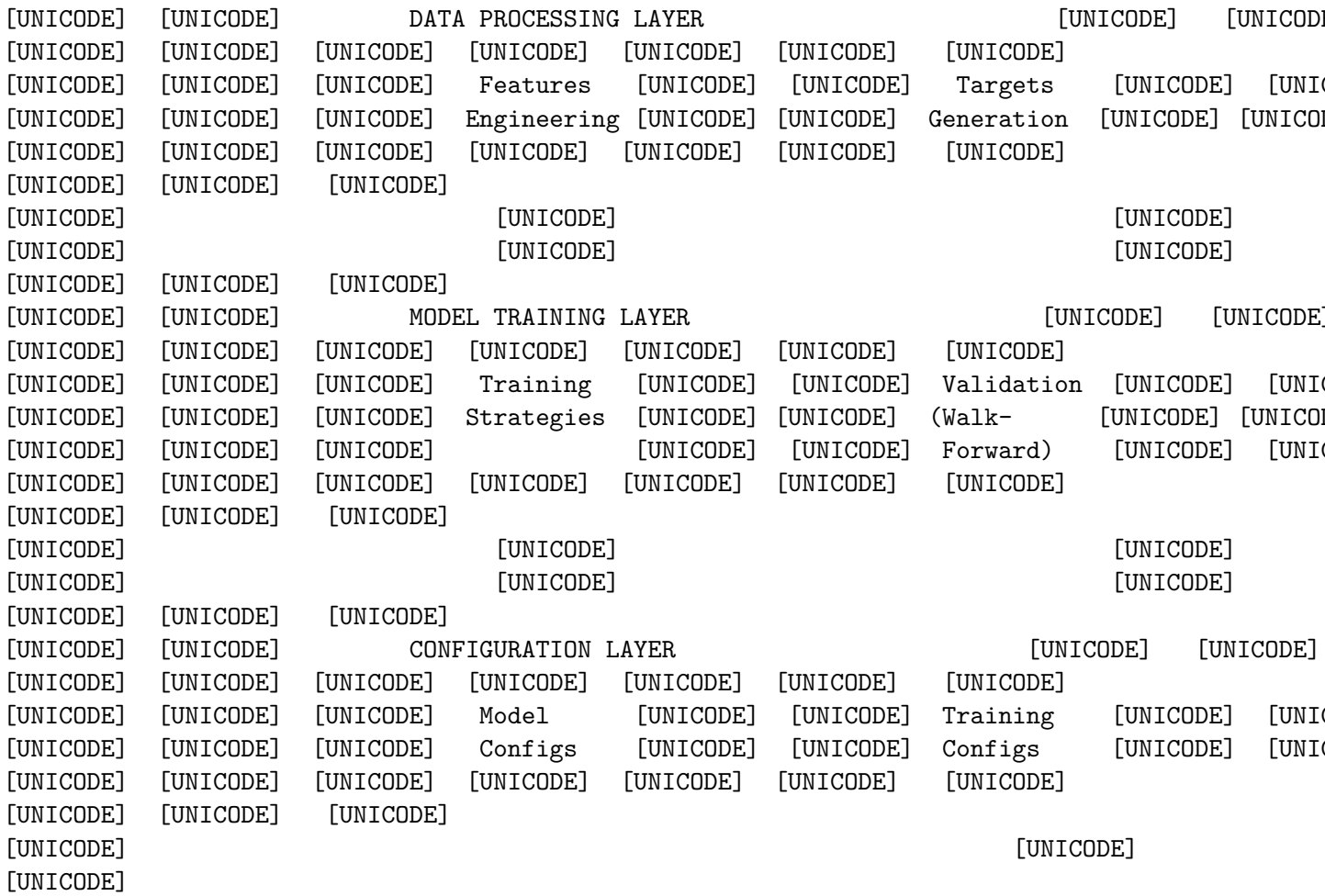
Key architectural principles: - **Client-hosted** [UNICODE] Software runs entirely on client infrastructure - **Modular design** [UNICODE] Clear separation of concerns across modules - **Configuration-driven** [UNICODE] All runtime parameters from centralized configs - **Pipeline-based** [UNICODE] Data flows through well-defined pipeline stages - **Extensible** [UNICODE] Easy to add new models, features, and strategies

---

2. High-Level Architecture

2.1 System Overview

```
[UNICODE]
[UNICODE]                    CLIENT INFRASTRUCTURE                          [UNICODE]
[UNICODE]                    (Client-Controlled)                           [UNICODE]
[UNICODE]
                              [UNICODE]
                              [UNICODE]
        [UNICODE]
        [UNICODE]                                                  [UNICODE]
        [UNICODE]                                                  [UNICODE]
[UNICODE]                       [UNICODE]
[UNICODE]  DATA ACQUISITION[UNICODE]                       [UNICODE]  DEPLOYMENT &    [UNICODE]
[UNICODE]  & INGESTION     [UNICODE]                       [UNICODE]  EXECUTION       [UNICODE]
[UNICODE]                  [UNICODE]                       [UNICODE]                  [UNICODE]
[UNICODE]  [UNICODE] Market Data   [UNICODE]                      [UNICODE]  [UNICODE] Model Se
[UNICODE]  [UNICODE] Data Sources  [UNICODE]                      [UNICODE]  [UNICODE] Trading
[UNICODE]  [UNICODE] Normalization [UNICODE]                      [UNICODE]  [UNICODE] Monitori
[UNICODE]                       [UNICODE]
         [UNICODE]                                           [UNICODE]
         [UNICODE]                                           [UNICODE]
         [UNICODE]                                           [UNICODE]
[UNICODE]
[UNICODE]              FOX ML INFRASTRUCTURE PLATFORM                    [UNICODE]
[UNICODE]              (Client-Hosted Software)                           [UNICODE]
[UNICODE]
[UNICODE]                                                                 [UNICODE]
[UNICODE]  [UNICODE]   [UNICODE]
```



---

3. Component Architecture

3.1 Data Processing Layer

Purpose: Transform raw market data into ML-ready features and targets.

Components:

```
DATA_PROCESSING/
[UNICODE] features/
[UNICODE]   [UNICODE] SimpleFeatureComputer      # Basic technical indicators
[UNICODE]   [UNICODE] ComprehensiveFeatureBuilder # 200+ feature engineering
[UNICODE]   [UNICODE] StreamingFeatureBuilder    # Streaming/online features
[UNICODE] targets/
[UNICODE]   [UNICODE] BarrierTargetBuilder       # Barrier-based targets
[UNICODE]   [UNICODE] ExcessReturnsBuilder      # Excess return targets
[UNICODE]   [UNICODE] HFTForwardReturnsBuilder  # High-frequency targets
[UNICODE] pipeline/
[UNICODE]   [UNICODE] DataNormalization          # RTH alignment, grid correction
[UNICODE]   [UNICODE] FeaturePipeline           # End-to-end feature workflow
```

```
[UNICODE]   [UNICODE] TargetPipeline            # End-to-end target workflow
[UNICODE] utils/
    [UNICODE] MemoryManagement          # Efficient memory handling
    [UNICODE] Logging                   # Structured logging
    [UNICODE] Validation                # Data sanity checks
```

Data Flow:

```
Raw Market Data
    [UNICODE]
Normalization (RTH alignment, grid correction)
    [UNICODE]
Feature Engineering (200+ technical features)
    [UNICODE]
Target Generation (barrier, excess returns, HFT)
    [UNICODE]
Labeled Dataset (features + targets)
```

3.2 Model Training Layer

Purpose: Train and validate ML models using walk-forward validation.

Components:

```
TRAINING/
[UNICODE] model_fun/
[UNICODE]   [UNICODE] LightGBMTrainer           # Gradient boosting
[UNICODE]   [UNICODE] XGBoostTrainer            # Gradient boosting
[UNICODE]   [UNICODE] MLPTrainer                # Deep learning
[UNICODE]   [UNICODE] TransformerTrainer        # Transformer models
[UNICODE]   [UNICODE] EnsembleTrainer           # Model ensembles
[UNICODE]   [UNICODE] MultiTaskTrainer          # Multi-task learning
[UNICODE]   [UNICODE] [13+ additional trainers] # Probabilistic, advanced models
[UNICODE] strategies/
[UNICODE]   [UNICODE] SingleTaskStrategy        # One model per target
[UNICODE]   [UNICODE] MultiTaskStrategy         # Shared model for targets
[UNICODE]   [UNICODE] CascadeStrategy           # Sequential dependencies
[UNICODE] walkforward/
[UNICODE]   [UNICODE] WalkForwardValidator      # Time-series validation
[UNICODE]   [UNICODE] PurgedTimeSeriesSplit     # Leakage-safe splitting
[UNICODE] memory/
    [UNICODE] MemoryManager             # Memory optimization
    [UNICODE] BatchProcessing           # Efficient batch handling
```

Training Flow:

```
Labeled Dataset
    [UNICODE]
Walk-Forward Validation (time-series split)
```

```
    [UNICODE]
Model Training (17+ model types)
    [UNICODE]
Model Validation (performance metrics)
    [UNICODE]
Trained Models (serialized)
```

3.3 Configuration Layer

Purpose: Centralized, version-controlled configuration management.

Components:

```
CONFIG/
[UNICODE] model_config/
[UNICODE]   [UNICODE] lightgbm.yaml             # LightGBM configs (3 variants)
[UNICODE]   [UNICODE] xgboost.yaml              # XGBoost configs (3 variants)
[UNICODE]   [UNICODE] mlp.yaml                  # MLP configs (3 variants)
[UNICODE]   [UNICODE] [14+ additional configs]  # All model types
[UNICODE] training_config/
[UNICODE]   [UNICODE] walkforward.yaml          # Walk-forward parameters
[UNICODE]   [UNICODE] feature_selection.yaml    # Feature selection configs
[UNICODE]   [UNICODE] first_batch_specs.yaml    # Training specifications
[UNICODE] config_loader.py
    [UNICODE] load_model_config()       # Load model configs
    [UNICODE] list_available_configs()  # List available configs
    [UNICODE] apply_overrides()         # Runtime overrides
```

Configuration Flow:

```
Base Config (YAML)
    [UNICODE]
Variant Selection (conservative/balanced/aggressive)
    [UNICODE]
Runtime Overrides (environment variables, CLI args)
    [UNICODE]
Final Configuration (validated, applied)
```

3.4 Model Output & Artifacts Layer

Purpose: Generation and storage of trained models, predictions, and performance metrics.

Components:

```
outputs/                          # Model artifacts and outputs
[UNICODE] models/                       # Serialized trained models
[UNICODE]   [UNICODE] lightgbm/                # LightGBM model artifacts
[UNICODE]   [UNICODE] xgboost/                 # XGBoost model artifacts
```

```
[UNICODE]   [UNICODE] [other model families]   # Additional model types
[UNICODE] predictions/                 # Model predictions
[UNICODE]   [UNICODE] validation/              # Validation set predictions
[UNICODE]   [UNICODE] test/                    # Test set predictions
[UNICODE] reports/                     # Performance reports
    [UNICODE] metrics/                 # Performance metrics
    [UNICODE] diagnostics/             # Model diagnostics
```

Output Flow:

```
Trained Models
    [UNICODE]
Model Serialization (standard formats)
    [UNICODE]
Signal Generation (trading signals)
    [UNICODE]
Risk Management (safety guards, position sizing)
    [UNICODE]
Order Execution (broker integration)
    [UNICODE]
Performance Tracking (PnL, metrics)
```

---

4. Data Flow Architecture

4.1 End-to-End Pipeline

```
[UNICODE]
[UNICODE]                    STAGE 1: DATA INGESTION                   [UNICODE]
[UNICODE]                                                               [UNICODE]
[UNICODE]  Raw Market Data (OHLCV)                                      [UNICODE]
[UNICODE]  [UNICODE] yfinance, custom data sources                              [UNICODE]
[UNICODE]  [UNICODE] Multiple timeframes (1m, 5m, 15m, 1h, 1d)                 [UNICODE]
[UNICODE]  [UNICODE] Multiple symbols (SPY, QQQ, IWM, etc.)                     [UNICODE]
[UNICODE]
                      [UNICODE]
                      [UNICODE]
[UNICODE]
[UNICODE]              STAGE 2: DATA PROCESSING                          [UNICODE]
[UNICODE]                                                                [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]    [UNICODE]  [UNICODE]
[UNICODE]  [UNICODE] Normalization[UNICODE] [UNICODE]  [UNICODE]   Feature    [UNICODE] [UNICOD
[UNICODE]  [UNICODE]              [UNICODE]    [UNICODE]  Engineering [UNICODE]    [UNICODE]  C
[UNICODE]  [UNICODE] [UNICODE] RTH align  [UNICODE]    [UNICODE]              [UNICODE]    [UNI
[UNICODE]  [UNICODE] [UNICODE] Grid corr  [UNICODE]    [UNICODE] [UNICODE] 200+ feat  [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] Clean data [UNICODE]    [UNICODE] [UNICODE] Technical  [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]    [UNICODE]  [UNICODE]
[UNICODE]
                      [UNICODE]
```

```
                        [UNICODE]
[UNICODE]
[UNICODE]               STAGE 3: MODEL TRAINING                          [UNICODE]
[UNICODE]                                                                 [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]    [UNICODE]  [UNICODE]
[UNICODE]  [UNICODE] Walk-Forward [UNICODE] [UNICODE]  [UNICODE]    Model     [UNICODE] [UNICOD
[UNICODE]  [UNICODE]  Validation  [UNICODE]    [UNICODE]   Training   [UNICODE]    [UNICODE]
[UNICODE]  [UNICODE]              [UNICODE]    [UNICODE]              [UNICODE]    [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] Time-split [UNICODE]    [UNICODE] [UNICODE] 17+ models [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] Purged     [UNICODE]    [UNICODE] [UNICODE] Strategies [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]    [UNICODE]  [UNICODE]
[UNICODE]
                        [UNICODE]
                        [UNICODE]
[UNICODE]
[UNICODE]               STAGE 4: DEPLOYMENT (Optional)                   [UNICODE]
[UNICODE]                                                                 [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]    [UNICODE]  [UNICODE]
[UNICODE]  [UNICODE]   Model      [UNICODE] [UNICODE]  [UNICODE]    Signal    [UNICODE] [UNICOD
[UNICODE]  [UNICODE]  Inference   [UNICODE]    [UNICODE]  Generation  [UNICODE]    [UNICODE]  L
[UNICODE]  [UNICODE]              [UNICODE]    [UNICODE]              [UNICODE]    [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] Real-time  [UNICODE]    [UNICODE] [UNICODE] Blending   [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] C++ engine [UNICODE]    [UNICODE] [UNICODE] Risk mgmt  [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]    [UNICODE]  [UNICODE]
[UNICODE]
```

---

5. Module Dependencies

5.1 Dependency Graph

```
CONFIG (Configuration Management)
    [UNICODE]
    [UNICODE] (used by)
    [UNICODE]
DATA_PROCESSING (Feature Engineering)
    [UNICODE]
    [UNICODE] (produces)
    [UNICODE]
TRAINING (Model Training)
    [UNICODE]
    [UNICODE] (uses)
    [UNICODE]
CONFIG (Model Configs)
    [UNICODE]
    [UNICODE] (optional)
    [UNICODE]
External Trading Systems (Optional Integration)
```

5.2 External Dependencies

Core Dependencies: - **Python 3.9+** [UNICODE] Core runtime - **NumPy, Pandas** [UNICODE] Data processing - **Scikit-learn** [UNICODE] ML utilities - **LightGBM, XGBoost** [UNICODE] Gradient boosting models - **PyTorch** [UNICODE] Deep learning models (optional)

Trading Dependencies: - **ib_insync** [UNICODE] Interactive Brokers API - **alpaca-trade-api** [UNICODE] Alpaca API

Infrastructure: - **Git** [UNICODE] Version control - **GitHub** [UNICODE] Repository hosting - **YAML** [UNICODE] Configuration format

---

6. Deployment Architecture

6.1 Client-Hosted Deployment

```
[UNICODE]
[UNICODE]                    CLIENT INFRASTRUCTURE                    [UNICODE]
[UNICODE]                                                              [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]              [UNICODE]
[UNICODE]  [UNICODE]   Development    [UNICODE]    [UNICODE]    Production    [UNICODE]
[UNICODE]  [UNICODE]   Environment    [UNICODE]    [UNICODE]    Environment   [UNICODE]
[UNICODE]  [UNICODE]                   [UNICODE]    [UNICODE]                  [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] Code repo      [UNICODE]    [UNICODE] [UNICODE] Model serving
[UNICODE]  [UNICODE] [UNICODE] Training       [UNICODE]    [UNICODE] [UNICODE] Trading      [U
[UNICODE]  [UNICODE] [UNICODE] Testing        [UNICODE]    [UNICODE] [UNICODE] Monitoring    [U
[UNICODE]  [UNICODE]    [UNICODE]              [UNICODE]
[UNICODE]                                                              [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]              [UNICODE]
[UNICODE]  [UNICODE]   Data Storage   [UNICODE]    [UNICODE]   Model Storage  [UNICODE]
[UNICODE]  [UNICODE]                   [UNICODE]    [UNICODE]                  [UNICODE]
[UNICODE]  [UNICODE] [UNICODE] Raw data       [UNICODE]    [UNICODE] [UNICODE] Trained models
[UNICODE]  [UNICODE] [UNICODE] Processed data [UNICODE]    [UNICODE] [UNICODE] Configs
[UNICODE]  [UNICODE] [UNICODE] Features       [UNICODE]    [UNICODE] [UNICODE] Results
[UNICODE]  [UNICODE]    [UNICODE]              [UNICODE]
[UNICODE]                                                              [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]              [UNICODE]
[UNICODE]  [UNICODE]   Compute        [UNICODE]    [UNICODE]   Networking     [UNICODE]
[UNICODE]  [UNICODE]                   [UNICODE]    [UNICODE]                  [UNICODE]
[UNICODE]  [UNICODE] CPU/GPU          [UNICODE]    [UNICODE] [UNICODE] Market data    [UNICODE]
[UNICODE]  [UNICODE] Memory           [UNICODE]    [UNICODE] [UNICODE] Broker APIs    [UNICODE]
[UNICODE]  [UNICODE] Storage          [UNICODE]    [UNICODE] [UNICODE] Internal       [UNICODE]
[UNICODE]  [UNICODE]    [UNICODE]              [UNICODE]
[UNICODE]
```

6.2 No Vendor Infrastructure

Important: Fox ML Infrastructure does not operate: - [ERROR] Vendor-hosted servers - [ERROR] Vendor-hosted databases - [ERROR] Vendor cloud services - [ERROR] Vendor network infrastructure

All infrastructure is client-controlled and client-managed.

---

7. Security Architecture

7.1 Security Boundaries

```
[UNICODE]
[UNICODE]                    VENDOR BOUNDARY                              [UNICODE]
[UNICODE]                                                                 [UNICODE]
[UNICODE]  [UNICODE] Code repositories (GitHub)                                 [UNICODE]
[UNICODE]  [UNICODE] Support communications (Email)                             [UNICODE]
[UNICODE]
                              [UNICODE]
                              [UNICODE] (Code Delivery)
                              [UNICODE]
                              [UNICODE]
[UNICODE]
[UNICODE]                    CLIENT BOUNDARY                              [UNICODE]
[UNICODE]                                                                 [UNICODE]
[UNICODE]  [UNICODE] Software execution                                         [UNICODE]
[UNICODE]  [UNICODE] Data processing                                            [UNICODE]
[UNICODE]  [UNICODE] Model training                                             [UNICODE]
[UNICODE]  [UNICODE] Trading operations                                         [UNICODE]
[UNICODE]  [UNICODE] All client data                                            [UNICODE]
[UNICODE]
```

7.2 Security Controls

Vendor-side: - Repository access controls (2FA, SSH keys) - Code integrity (version control, signed commits) - Supply chain integrity (no telemetry, explicit dependencies)

Client-side: - Infrastructure security (client-managed) - Data security (client-managed) - Network security (client-managed) - Access control (client-managed)

---

8. Scalability Architecture

8.1 Horizontal Scalability

Scalable components: - **Data processing** [UNICODE] Parallel processing of multiple symbols/timeframes - **Model training** [UNICODE] Distributed training (if configured) - **Feature engineering** [UNICODE] Batch processing for large datasets - **Inference** [UNICODE] Parallel inference for multiple models

8.2 Vertical Scalability

Resource requirements: - **CPU** [UNICODE] Multi-core processors (8+ cores recommended) - **RAM** [UNICODE] 16GB+ (32GB+ for large datasets) - **GPU** [UNICODE] Optional but recommended for deep learning - **Storage** [UNICODE] Sufficient for datasets and models

---

9. Integration Points

9.1 Data Integration

Data sources: - **yfinance** [UNICODE] Equity market data (default) - **Custom adapters** [UNICODE] Client-specific data sources - **File-based** [UNICODE] CSV, Parquet, HDF5 files

9.2 Model Integration

Model deployment: - **Serialized models** [UNICODE] Pickle, joblib, PyTorch formats - **REST API** [UNICODE] Model serving via REST API (if configured)

9.3 Model Integration

Integration points: - Models can be integrated with external trading systems - Standard formats for model serialization - Performance tracking and monitoring interfaces

---

10. Configuration Architecture

10.1 Configuration Hierarchy

```
Base Config (YAML)
    [UNICODE]
Variant Selection (conservative/balanced/aggressive)
    [UNICODE]
Environment Overrides (environment variables)
    [UNICODE]
Runtime Overrides (CLI arguments)
    [UNICODE]
Final Configuration (validated, applied)
```

10.2 Configuration Sources

Configuration sources: - **YAML files** [UNICODE] `CONFIG/model_config/*.yaml` - **Environment variables** [UNICODE] Runtime overrides - **CLI arguments** [UNICODE] Command-line overrides - **Code defaults** [UNICODE] Fallback defaults

---

11. Monitoring and Observability

11.1 Logging

Logging components: - **Structured logging** [UNICODE] JSON-structured logs - **Log levels** [UNICODE] DEBUG, INFO, WARNING, ERROR - **Contextual information** [UNICODE] Run IDs, symbols, fold numbers - **Client-managed** [UNICODE] Logs stored and managed by client

11.2 Metrics

Metrics tracked: - **Performance metrics** [UNICODE] Training time, inference latency - **Model metrics** [UNICODE] Accuracy, loss, validation scores - **Pipeline metrics** [UNICODE] Data processing throughput - **Client-managed** [UNICODE] Metrics collected and stored by client

---

12. Contact

For architecture questions or integration planning:

Jennifer Lewis
Fox ML Infrastructure LLC
Email: **jenn.lewis5789@gmail.com**
Subject: *Architecture Inquiry [UNICODE] Fox ML Infrastructure*

---

13. Related Documents

- `docs/00_executive/ARCHITECTURE_OVERVIEW.md` [UNICODE] Detailed architecture overview
- `docs/03_technical/design/ARCHITECTURE_DEEP_DIVE.md` [UNICODE] Deep technical dive
- `LEGAL/ENTERPRISE_DELIVERY.md` [UNICODE] Repository structure and delivery model
- `LEGAL/CLIENT_ONBOARDING.md` [UNICODE] Client onboarding and integration guide

---

14. Summary

Key Architectural Principles:

1. **Client-hosted** [UNICODE] Software runs entirely on client infrastructure
2. **Modular** [UNICODE] Clear separation of concerns across modules
3. **Configuration-driven** [UNICODE] All runtime parameters from configs
4. **Pipeline-based** [UNICODE] Data flows through well-defined stages
5. **Extensible** [UNICODE] Easy to add new models, features, strategies
6. **Scalable** [UNICODE] Horizontal and vertical scalability
7. **Secure** [UNICODE] Security boundaries and controls clearly defined

This architecture provides a comprehensive foundation for ML research and quantitative workflows.