**Design and Technical Documentation**

**1. Purpose and Scope**

1. **Project Background**
   * Background: current risk management systems in financial markets rely heavily on historical trading data and real-time intraday monitoring, but are unable to anticipate in advance the impact of pre-opening news on trading volumes.
   * Overall goal of the project: Helps DTCC dynamically adjust margin requirements and clearing resources
2. **Document Objectives**
   * Provide a technical reference for design and implementation to facilitate development, maintenance, and presentation.
   * Define system architecture, components, data flow, and algorithm principles.

**2. System Architecture Overview**

1. **High-Level System Description**
   * Major modules: data ingestion, data preprocessing, data storage, sentiment analysis, model training, prediction service, front-end/outputs
   * Architecture Diagram:

A diagram of a data flow

AI-generated content may be incorrect.

1. **Technology Stack**
   * Data collection tools or languages: Python scripts, APIs(Refinitiv Eikon, Interactive Brokers, Yahoo Finance)
   * Data storage solutions: SQL database
   * Model development environment: Python + scikit-learn/Hugging Face Transformers
   * Front-end or visualization tools: Streamlit
2. **Module Breakdown and Responsibilities**
   * Data Ingestion: Collect news headline and stock volume data.
   * Data Preprocessing & Feature Engineering: Clean, merge, and transform data.
   * Model Training & Validation: sentiment analysis and train volume prediction models
   * Model Serving: Deploy the models as APIs or services.

**3. Data Pipeline**

1. **Data Sources & Collection**
   * News Data: APIs (Refinitiv Eikon, Interactive Brokers), time range (2024.01.01-2024.12.31)
   * Stock Market Data: APIs (Yahoo Finance) for volume
2. **Data Storage & Management**
   * Database: SQL database
   * File formats: CSV
3. **Data Preprocessing**
   * Text Cleaning: Removing duplicates, noise, tokenization
   * Data Alignment: Synchronizing sentiment scores and volume time series.
   * Feature Construction: Lag features, Sentiment Score, Options Expiration Day, Month open/ close, Day-of-week, normalization
4. **Data Quality Control**
   * Ensuring data completeness and consistency.
   * Handling missing or anomalous values.

**4. Model Design and Algorithm Details**

* 1. **Sentiment Analysis Model**

1. **Model Selection**
   * Pre-trained models used: FinBERT, Vander, Roberta(Selected)
2. **Sentiment Scoring Process**
   * Input: News Headline.
   * Output: news count, sentiment score (ranging from -1 to 1), absolute sentiment score (ranging from 0 to 1) and standard deviation.
3. **Quality Evaluation**
   * Validation datasets or manual review process.

**4.2 Volume Prediction Model**

1. **Objectives and Features**
   * Target: Volume during the first hour of trading (9:30–10:30).
   * Features:

A black screen with white lines

AI-generated content may be incorrect.

1. **Model Type**
   * Algorithms used: Regression (Linear, XGBoost, Random Forest) and time series (SARIMAX)
   * Rationale for choice (accuracy, speed, suitability for time series data).
2. **Training & Validation**
   * Train data (2024.01.01-2024.09.30)/test data (2024.10.01-2024.12.31)
   * Evaluation metrics (MSE, RMSE, MAE, R-Squared)
3. **Hyperparameter Tuning**
   * Methods (manual tuning, grid search).
   * Key parameters (MSE, R-squared, learning rate)
4. **Final Model Selection & Performance**
   * Model comparison. (MSE, R-squared)
   * Test set results and why the chosen model is preferred.

**5. Implementation Details**

1. **Code Structure**
   * Folder hierarchy:

* data/: Stores raw data files (e.g., .csv) related to news, volume, or other relevant datasets.
* models/: Contains scripts that define and train the prediction models.
* data\_process/: Holds Jupyter notebooks (\*.ipynb) used for data cleaning, sentiment analysis, feature engineering, and exploratory analysis.
  + Main scripts:
* rfinitive.ipynb: Fetches news data and does basic checks
* sentiment\_analysis\_roberta.ipynb: Performs RoBERTa-based sentiment analysis on news.
* data\_engineering\_2.ipynb: Merges sentiment scores with volume data and creates new features
  + Main modules: Linear Regression, XGBoost, SARIMAX, Random Forest

1. **Environment & Dependencies**
   * Language versions: Python 3.12
   * Key libraries: pandas, numpy, scikit-learn, transformers, xgboost, statsmodels

**6. Deployment and Delivery**

1. **Deployment Environment**
   * Hosted Deployment: The application is deployed on a web server, and users can access it directly via a URL.
2. **Service/UI Design**
   * Accessible via a fully hosted web application.
   * Input/Output: Users interact with the application through a simple graphical user interface (GUI) in their web browser.
   * Steps for Users:

* Open the provided URL in any modern web browser:

https://liquidityguards-bluedevils.streamlit.app/

* Use the web interface to make selections.
  + Example Usage:
* Input: Users select a stock and choose a prediction model through the web interface
* Output: The results, including predictions and visualizations, are displayed on the webpage.

**7. Risks and Limitations**

1. **Project Risks**
   * Dependence on external news APIs or data providers.
   * Market volatility (unexpected events impacting predictions).
2. **Technical Constraints**
   * Model size vs. inference speed.
   * Infrastructure limitations (GPU availability, cloud costs).
3. **Business Constraints**
   * Prediction limited to a short time window.
   * Exclusion of other market factors not accounted for in the model.

**8. Future Work**

1. **Feature Expansion**
   * Integrate additional indicators such as technical, fundamental, and macroeconomic factors.
   * Extend predictions to multi-stock or multi-asset scenarios.
   * Include price forecasting capabilities.
2. **Model Enhancements**
   * Explore advanced NLP models or domain-specific variants for better performance.
   * Incorporate multi-modal data, such as social media sentiment and macroeconomic trends.
3. **Testing and Validation**
   * Develop comprehensive testing strategies, including unit tests, integration tests, and end-to-end tests.
   * Conduct performance testing for large-scale data processing.
   * Implement historical data backtesting to assess trading strategy impact.
4. **Automation & Intelligence**
   * Use AutoML for automated model selection and hyperparameter tuning.
   * Apply advanced deep learning techniques like LSTM or Transformers for time-series predictions.
5. **Monitoring and Maintenance**
   * Introduce monitoring metrics for real-time tracking of model performance and service uptime.
   * Set up alert mechanisms to notify about anomalies via Slack, email, or SMS.
   * Plan periodic model retraining and updates, either on a fixed schedule or based on performance thresholds.
   * Perform routine maintenance, such as data source stability checks and log reviews, to ensure system reliability.

**9. Materials**

1. **Repository Links**
   * GitHub: https://github.com/finos-labs/dtcch-2025-blue-devils
   * This repository contains the core source code, scripts for data processing, and model training notebooks.