Low-Level Design (LLD) Document Cryptocurrency Volatility Prediction System

**Contents**

1. [Component Breakdown](#_bookmark0) 2
   1. [Data Ingestion](#_bookmark1) 2
   2. [Data Preprocessing](#_bookmark2) 2
   3. [Exploratory Data Analysis (EDA)](#_bookmark3) 2
   4. [Model Training](#_bookmark4) 2
   5. [Model Evaluation](#_bookmark5) 3
   6. [Deployment](#_bookmark6) 3
2. [Data Flow](#_bookmark7) 3
3. [Error Handling](#_bookmark8) 3

# Component Breakdown

## Data Ingestion

* + - **Input**: CSV file (dataset.csv) with columns: date, crypto\_name, open, high, low, close, volume, marketCap.
    - **Implementation**:
      * Use pandas.read\_csv to load the dataset.
      * Validate data types and ensure no critical columns are missing.

## Data Preprocessing

* + - **Handle Missing Values**:
      * Drop rows with missing values in critical columns (open, high, low, close, vol- ume, marketCap).
      * Drop Unnamed: 0 column.
    - **Feature Engineering**:
      * Daily Return: *closet−closet−*1 .

*close −*1

*t*

* + - * Rolling Volatility (7-day): Standard deviation of daily returns over a 7-day win- dow, grouped by crypto\_name.
      * Liquidity Ratio:  *volume* .

*marketCap*

* + - * 7-day Moving Average: Mean of close prices over a 7-day window.
    - **Normalization**:
      * Apply StandardScaler to numerical features (open, high, low, close, volume, marketCap, daily\_return, rolling\_volatility\_7d, liquidity\_ratio, ma\_7\_close).

## Exploratory Data Analysis (EDA)

* + - **Summary Statistics**: Use df.describe() to compute mean, std, min, max, etc.
    - **Visualizations**:
      * Price trends: Line plot of close prices for Bitcoin.
      * Volatility: Line plot of 7-day rolling volatility for Bitcoin.
      * Liquidity: Line plot of liquidity ratio for Bitcoin.
    - **Output**: Save plots as PNG files (bitcoin\_close\_price.png, bitcoin\_volatility.png, bit- coin\_liquidity.png).

## Model Training

* + - **Model**: RandomForestRegressor with 100 trees, random\_state=42.
    - **Features**: open, high, low, close, volume, marketCap, daily\_return, rolling\_volatility\_7d, liquidity\_ratio, ma\_7\_close.
    - **Target**: rolling\_volatility\_7d (shifted for prediction).
    - **Splitting**: Use TimeSeriesSplit with 5 folds.
    - **Training**: Fit model on training data for each fold.

## Model Evaluation

* + - **Metrics**:
      * RMSE: √*mean*\_*squared*\_*error*(*ytest, ypred*)
      * MAE: mean\_absolute\_error(*ytest, ypred*)
      * R²: r2\_score(*ytest, ypred*)
    - **Implementation**: Compute metrics for each fold and average them.

## Deployment

* + - **Framework**: Streamlit
    - **Implementation**:
      * Save model and scaler using joblib.
      * Create input fields for features in Streamlit.
      * Normalize inputs, predict volatility, and display results.

# Data Flow

1. Load CSV *→* Validate *→* Preprocess (clean, engineer, normalize) *→* Save processed data.
2. Processed data *→* EDA (statistics, plots) *→* Save outputs.
3. Processed data *→* Train model *→* Evaluate *→* Save model and metrics.
4. Load model and scaler *→* Streamlit app *→* User inputs *→* Predict *→* Display.

# Error Handling

* Check for missing values and invalid types before processing.
* Handle division by zero in liquidity ratio calculation.
* Ensure no NaN values are passed to the model.