PIPELINE ARCHITECTURE

Data Flow Explanation (Preprocessing → Prediction)

The system follows a structured pipeline to ensure accurate, reproducible, and scalable volatility prediction.

Step-by-Step Data Flow

Raw Cryptocurrency Data

→ Data Preprocessing

→ Feature Engineering

→ Exploratory Data Analysis

→ Model Training

→ Model Evaluation

→ Prediction

1. Data Preprocessing

- Raw OHLC, volume, and market capitalization data are cleaned.
- Missing values in price data are handled using forward-fill techniques.
- Volume and market capitalization values are validated to remove invalid entries.
- Data is sorted chronologically for each cryptocurrency to preserve time-series integrity.

Output: Clean, consistent, time-ordered dataset.

---

2. Feature Engineering

- New features are created to capture market dynamics:
  - Log returns
  - High–Low price spread
  - Rolling volatility (14-day)
  - Liquidity ratio (Volume / Market Cap)
  - Moving averages (SMA, EMA)
- Rolling and lag-based calculations ensure only past data is used.

Output: Feature-enriched dataset suitable for modelling.

---

3. Exploratory Data Analysis (EDA)

- Visual and statistical analysis is performed to understand:
  - Volatility distribution
  - Price trends over time
  - Relationship between volume and volatility
  - Correlation between engineered features
- Insights from EDA guide feature selection and model choice.

Output: Analytical understanding and validated feature set.

---

4. Model Training

- The dataset is split using a time-based approach (train on past data, test on future data).
- Features are scaled to improve model stability.
- A regression-based machine learning model (XGBoost) is trained to predict volatility.

Output: Trained volatility prediction model.

---

5. Model Evaluation

- Model predictions are evaluated using:
  - RMSE
  - MAE
  - R^2 score
- Performance metrics quantify prediction accuracy and reliability.

Output: Validated model performance results.

---

6. Prediction & Deployment

- The trained model is deployed using a Flask API.
- Users send feature data via a POST request.
- The system returns predicted volatility in JSON format.

Output: Real-time volatility predictions.