📌 4. Pipeline Architecture + Full Documentation

A. Data Pipeline

1. Raw data ingestion
2. Cleaning
3. Feature engineering
4. Save engineered file
5. Train-test split
6. Feature scaling (optional)
7. Model training
8. Save model

B. Application Pipeline

User Input → Validation → Feature Selection → Model → Prediction → Report/Charts

---

C. Deployment Pipeline

- Streamlit app deploys locally
- Model loaded from /models
- Autofill loads from processed dataset

📌 5. Final Project Report

---

1. Project Title

Cryptocurrency Liquidity Prediction for Market Stability

---

2. Problem Statement

Financial markets require accurate liquidity estimation to ensure stability and trading efficiency.

This project predicts **liquidity_score**, which measures ease of trading a coin without price impact.

---

3. Objective

Build a prediction system that:

- Ingests real crypto market data
- Trains a regression model
- Predicts liquidity in real-time
- Offers reporting tools
- Provides interactive UI

---

4. Methodology

✓ Exploratory Data Analysis
✓ Feature Selection
✓ Train Linear Regression Model
✓ Build Streamlit front-end
✓ Provide Single & Batch Mode
✓ Logging + PDF + History

---

5. Model Used

- **Linear Regression**
- Selected because:
    - Simple, robust
    - Interpretable
    - Performs well with numeric features

---

6. Results

- MAE: *depends on dataset*
- Important features:

1. liquidity_ratio
2. 24h_volume
3. mkt_cap
4. price_change_24h

---

7. Deliverables

- Feature engineered dataset
- Trained model (.pkl)
- Streamlit App
- Reports (PDF, CSV)
- Documentation (HLD, LLD, Pipeline)

---

8. Conclusion

The application successfully predicts liquidity_score using only numeric features and provides a fully interactive UI with reports and logs. The system is stable, reproducible, and ready for real-world expansion.