Goal

Build a fully containerised mini-pipeline that streams live trade data for at least one Binance spot-symbol (e.g., *BTC-USDT*) into ClickHouse DB, using Redpanda as the Kafka-compatible broker.

Deliverables

1. **docker-compose.yaml** that starts all required services.

2. Producer code (redpanda connect pipeline preferred or language of election) that pulls trades from Binance's WebSocket stream and publishes JSON records to Redpanda.

3. A ClickHouse table fed from the topic (either via the Kafka table engine or a consumer process—your choice). Define partition and schema strategy.

4. A short **README** with architecture diagram, setup steps, test queries, and design decisions.

Time box ≈ 6–8 h of focused work; you may spread it over a few days.

Technology constraints

- Use Docker (+ docker-compose).
- Programming language: Python, Go, or Rust.
- Use only Free and Open Source Software (FOSS) packages.

What we score: code clarity, git hygiene, container orchestration, data model & quality checks, documentation.

Theoretical questions

Data-Quality & Testing

How would you detect and filter out both duplicate messages and null-field records in your streaming pipeline? Describe how you'd write automated tests (unit/integration) or use synthetic "bad" data to prove your logic works.

ClickHouse Materialized Views

What did you learn about using ClickHouse's Kafka table engine paired with a MATERIALIZED VIEW into a MergeTree table?

Architecture & Quality-Checkpoint Diagram

Draw or explain an end-to-end diagram showing the solution and any components you consider relevant (eg: table engine, redpanda topic, etc)

Helpful links

- **Redpanda Docker-Compose Quick-Start** – official docs (shows minimal compose file). https://docs.redpanda.com/current/get-started/quick-start/

- **ClickHouse Docker getting-started** – pulls latest stable image & notes ports. https://clickhouse.com/docs/install/docker

- **Binance WebSocket Stream docs** – trade stream endpoint format & reconnection rules. https://developers.binance.com/docs/binance-spot-api-docs/web-socket-streams