Shipping a Data Product: From Raw Telegram Data to an Analytical API

Project Overview

This document outlines the implementation of an end-to-end data pipeline for Telegram data analysis.
The project uses:
- Telethon for Telegram data scraping
- dbt for transformation in PostgreSQL
- YOLOv8 for image object detection
- FastAPI for exposing data insights via API
- Dagster for orchestration and scheduling

Dockerfile

```
FROM python:3.11-slim
WORKDIR /app
COPY requirements.txt ./
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
CMD ["bash"]
```

docker-compose.yml

```
version: '3.8'
services:
  postgres:
    image: postgres:15
    environment:
      POSTGRES_USER: user
      POSTGRES_PASSWORD: password
      POSTGRES_DB: telegram_db
    ports:
      - "5432:5432"
  api:
    build: .
    depends_on:
      - postgres
    ports:
      - "8000:8000"
    command: uvicorn api.main:app --reload --host 0.0.0.0 --port 8000
```

Scraping Script (Telethon)

```
from telethon.sync import TelegramClient
from telethon.tl.functions.messages import GetHistoryRequest
import os, json, datetime
client = TelegramClient('session', api_id, api_hash)

async def scrape_channel(channel):
    await client.start()
    entity = await client.get_entity(channel)
    history = await client(GetHistoryRequest(peer=entity, limit=100))
    # Save messages to JSON
```

Shipping a Data Product: From Raw Telegram Data to an Analytical API

YOLOv8 Detection

```
from ultralytics import YOLO
model = YOLO('yolov8n.pt')
# Loop through images and detect objects
```

FastAPI Endpoint Example

```
from fastapi import FastAPI
from api.crud import get_top_products
app = FastAPI()

@app.get("/api/reports/top-products")
def top_products(limit: int = 10):
    return get_top_products(limit)
```

Dagster Pipeline

```
from dagster import job, op

@op
def scrape(): pass
@op
def load(): pass
@op
def transform(): os.system("dbt run")
@op
def enrich(): os.system("python detect_objects.py")

@job
def telegram_pipeline():
    scrape() >> load() >> transform() >> enrich()
```