**🚀 Interim Submission Report**

**Challenge**: Week 7 – 10 Academy: Artificial Intelligence Mastery  
**Project Title**: Shipping a Data Product: From Raw Telegram Data to an Analytical API  
**Date**: 12 July 2024  
**Name**: Bisrat Haile

**📌 1. Project Overview and Understanding**

The core objective of this challenge is to build a **production-grade data product** that extracts Telegram data related to Ethiopian medical businesses, transforms it into a structured data warehouse, enriches it with computer vision, and makes it accessible via a clean, fast analytical API.

We aim to answer questions like:

* What are the most mentioned medical products or drugs?
* How does product availability or price differ across Telegram channels?
* Which channels post the most images?
* What are the temporal trends in health-related discussions?

To solve this, we’ve adopted a **modern ELT architecture** with the following stack:

* **Telethon** for Telegram data scraping
* **PostgreSQL** as our warehouse
* **dbt** for data transformation and modeling
* **YOLOv8** for object detection from images
* **FastAPI** for exposing insights as APIs
* **Dagster** for orchestration

**🔧 2. Methodology – Process, Progress So Far, and Results**

**✅ Step 0: Environment Setup**

* Set up a professional Git project
* Created requirements.txt for dependencies
* Added Docker containers for PostgreSQL and development
* Used .env + python-dotenv for secure secret management
* Verified all services with dbt debug and local database connection

**✅ Step 1: Telegram Data Scraping**

* Used Telethon to collect messages and images from channels like:
  + Chemed Telegram Channel
  + Lobelia4Cosmetics
  + Tikvah Pharma
* Saved structured raw JSON into partitioned folders:  
  data/raw/YYYY-MM-DD/channel\_name.json
* Added flags for messages with images
* Implemented logging and graceful handling of missing data or rate limits

**✅ Step 2: Data Modeling and Transformation (dbt)**

* Loaded raw JSON data into the PostgreSQL raw.telegram\_messages table
* Created the following dbt models:
  + stg\_telegram\_messages: staging layer to clean/normalize raw data
  + dim\_channels: unique Telegram channels (dimension table)
  + dim\_dates: calendar dimension generated from message dates
  + fct\_messages: fact table linking messages to channels and dates, with metrics like message\_length, has\_image, and image\_path
* Added dbt tests for not\_null and unique constraints
* Ensured all models are modular and materialized as view or table as appropriate
* All dbt runs successful as of this submission ✅

**⚠️ 3. Challenges & Solutions (Expanded)**

| **Challenge** | **Solution** |
| --- | --- |
| **PostgreSQL connection refused** | Installed psql client, ensured Docker service running, verified .env credentials |
| **profiles.yml not found** | Created the correct ~/.dbt/profiles.yml file with matching profile name in dbt\_project.yml |
| **Missing folders or files** | Carefully restructured project folders to match expected paths for dbt and Python scripts |
| **FileNotFoundError when loading JSON** | Implemented proper path handling and fallback logging |
| **dbt parsing errors** | Fixed YAML indentation and removed invalid dbt\_project.yml from nested model directories |
| **Git commit confusion** | Used clear commit messages after each feature milestone |

**📅 4. Future Plan – What’s Left and How I Plan to Finish**

**✅ Completed:**

* Raw data ingestion
* Database modeling and transformation
* Core analytics-ready star schema

**🔜 In Progress / To Do:**

| **Task** | **Description** | **Estimated Time** |
| --- | --- | --- |
| **Task 3: YOLOv8 Enrichment** | Run YOLOv8 on images from scraped messages. Create fct\_image\_detections table. | 1 day |
| **Task 4: FastAPI Analytics** | Create REST API endpoints using FastAPI (top products, channel activity, search). | 1 day |
| **Task 5: Dagster Orchestration** | Build a Dagster pipeline to orchestrate all stages and schedule daily jobs. | 1 day |

I'll prioritize YOLO enrichment next, followed by a FastAPI layer built directly on the cleaned dbt models. After this, I’ll finalize everything into a Dagster job that runs the entire pipeline from raw data to deployed API.

**✅ 5. Conclusion – Summary of Progress and Confidence**

This project has been a comprehensive hands-on experience in shipping real-world data products. So far, I have successfully:

* Set up a modern data environment
* Ingested and structured raw Telegram data
* Modeled and tested a robust star schema using dbt
* Managed all components inside a reproducible project

I now have full confidence in completing the enrichment, API, and orchestration in time for final submission.

This challenge has deepened my skills in dbt, ELT pipeline thinking, and cross-functional system design — bridging scraping, transformation, modeling, and deployment.

**🔜 6. Moving Forward**

* **Immediate focus**: YOLOv8 image detection and loading detections into a linked fact table
* **Midterm focus**: Build a fully functional FastAPI app with real query endpoints
* **Final steps**: Wrap all pieces into a unified Dagster pipeline, validate, and document
* **Stretch goals**: Add Docker Compose support for the full stack + Postman collection for API testing