Telegram ETL Pipeline

Overview:

This project proposes building an ETL (Extract, Transform, Load) pipeline using a Telegram bot as the interface to gather, process, and load data. The pipeline will leverage Large Language Models (LLMs) for data transformation and analysis. Users will interact with the bot through various commands to submit data, extract insights, and receive processed output. The bot will provide a convenient interface for users, while the backend will handle data processing and transformation using LLMs.

Telegram Bot Commands:

1. **/start**
    - **Description:** Initializes the bot and provides an introduction to the user.
    - **Action:** Sends a welcome message, explaining the bot's purpose and listing the available commands. The bot will ask the user for their permission to interact and collect data for processing.
2. **/help**
    - **Description:** Provides a list of commands and their functionality.
    - **Action:** Displays detailed instructions on how to use the bot, including descriptions of the ETL process and how LLM models will assist with data analysis and transformation.
3. **/submitdata**
    - **Description:** Allows users to submit data to the ETL pipeline for extraction and processing.
    - **Action:** Prompts the user to upload a file (text, CSV, JSON, etc.) or input data directly. Once the data is submitted, it is extracted and stored for processing.
    - **Backend Process:** The data is parsed and cleaned before being sent to the transformation phase, where LLM models will analyze and extract key insights.
4. **/transformeddata**
    - **Description:** Retrieves the transformed data after processing by LLM models.
    - **Action:** Sends the user the processed data based on their input. Users can choose from different formats (summary, sentiment analysis, structured data, etc.).
    - **Backend Process:** After extraction, the LLM models will transform the data—this could involve summarization, sentiment classification, or data restructuring.
5. **/load**
    - **Description:** Loads the processed data to a specific database or destination.

- **Action:** Sends the user a confirmation once the transformed data has been successfully loaded into a database (e.g., SQL, NoSQL) or exported as a file (e.g., CSV, JSON).
- **Backend Process:** The processed data is loaded into the specified storage system or shared with external APIs for further use.

6. **/analyze [model]**
   - **Description:** Allows users to analyze submitted data using different LLM models (e.g., sentiment analysis, topic modeling, etc.).
   - **Action:** Users can specify a model (e.g., GPT) for analyzing the data. The bot will respond with the output of the analysis.
   - **Backend Process:** Depending on the selected model, the bot runs the data through various pre-trained or fine-tuned LLM models and returns the analysis result.
7. **/status**
   - **Description:** Provides the status of the ETL pipeline, including the current stage of processing.
   - **Action:** Sends real-time updates to the user about whether the data is still in extraction, transformation, or loading phase.
   - **Backend Process:** The bot queries the ETL pipeline to check the status of the user's data and returns a progress report.
8. **/history**
   - **Description:** Retrieves a list of the user’s previous interactions with the bot, including submitted data and processed results.
   - **Action:** Displays a history of data submitted by the user, along with a summary of the transformations applied.
   - **Backend Process:** Queries the database for a history of user interactions and previously processed datasets.
9. **/delete [id]**
   - **Description:** Allows users to delete a specific dataset from the pipeline.
   - **Action:** After receiving the ID of the dataset, the bot removes it from the system and confirms deletion.
   - **Backend Process:** Removes the dataset from both temporary storage and the final database, ensuring the data is no longer processed.
10. **/feedback**
    - **Description:** Collects feedback from users regarding the pipeline or bot experience.
    - **Action:** Prompts the user to submit feedback or rate their experience. This data will be analyzed for further improvements to the system.
    - **Backend Process:** Stores feedback in a database and optionally applies sentiment analysis using an LLM to gauge user satisfaction.
11. **/config**
    - **Description:** Allows users to configure the ETL process, such as setting the output format or changing LLM models.
    - **Action:** Prompts the user to update their preferences for output formats (e.g., JSON, CSV) and choose which models should be used for data transformation or analysis.
    - **Backend Process:** Updates the user's configuration settings in the database, ensuring future data processing follows the user’s preferences.

ETL Pipeline Flow:

1. **Extract:**
    - Users submit data through the `/submitdata` command. The bot validates the data and ensures it is in the correct format for processing.
2. **Transform:**
    - Data is passed through LLM models for transformation. This may include summarization, classification, or analysis, depending on the use case specified by the user.
3. **Load:**
    - Processed data is loaded into a destination specified by the user, such as a database or an external system.

Use of LLM Models:

- LLMs will be integrated into the transformation phase, handling tasks such as text summarization, sentiment analysis, and data classification.
- Users will have the flexibility to select which model to use for specific data transformations, allowing for customizable outputs.

Additional Features:

- **Error Handling:** The bot will notify users in case of any issues during the ETL process and provide suggestions for troubleshooting.
- **Data Security:** All interactions with the bot will be secured, and data will be encrypted during transmission and storage.