Task: Job titles grouping (supervised, unsupervised and/or semi-supervised learning)

You are given a dataset about raw job titles text. This technical task outlines the process of creating a code pipeline (Python) to extract job levels (Student, Junior Specialist, Assistant, Specialist, Sr Specialist, Manager, Director, VP, C-Level, Founder/Owner) and job areas (according to NAICS classification) from them.

Data - https://www.kaggle.com/datasets/jatinchawda/job-titles-and-description

- The dataset consists of a single column:
    - **job_title**: the raw form job title.

Task 1:

Features Creation:

- Create embeddings for the raw titles using any best-practice text representation methods (eg. pre-trained model embeddings from Hugging Face or GPT embeddings through API calls).
  Note: feel free to proceed with sampled data only. Apply basic data (text) preprocessing if you find necessary.

Task 2:

Model Building:

- Develop algorithm(s) that extract job level, job area groups (as described above) through named entity recognition, keyword extraction, or any other preferable technique.
- Split the data into several chunks. Predict the groups for the first chunk as identified in the previous step, review/assess the results to (possibly) use in the next steps (eg. build a classifier suitable for this task).

Task 3 (nice to have)**:**

For deployment:

- Wrap up your algorithm into one or several *.py* scripts with auxiliary files (if necessary). The main script should be customizable, taking the following parameters:
  – **path_to_input_file** - path to the input .CSV file
  – **path_to_output_file** - path where to save the processed .CSV file
- Write an appropriate Dockerfile that results in the execution of the script. Write a README explaining the steps (how to build an image given the Dockerfile, how to execute, etc.).

Constraints:

- Time limit: **5 business days**.

- The code should be written in Python using libraries like scikit-learn, Pandas, and other standard tools. If additional libraries (e.g., Hugging Face or Azure for API calls) are used, it is also welcomed.

Expected Results:

Upon completion of this task, the following deliverables are expected:

- Code that performs all steps (either a Jupyter notebook or archive of all scripts, based on your selected approach).
- A brief explanation of the chosen approach and models, decisions/optimizations (eg. scalability/code optimization technique for big amounts of data) made and the results (can be a markdown in the notebook or a separate pdf file).