MLOps Engineer Take-Home Assignment

You will have 3 days to complete the following assessment. You may submit at any time before the deadline.

If you notice any errors or have any questions, please reach out.

Please note, any scenarios presented below are fictional and the data has been fabricated for the purpose of this exercise.

Scenario

You've recently joined the MLOps team at Protex AI. Your task is to help productionise an early-stage computer vision dataset generation pipeline. The initial version is used internally by the CV team to generate and pre-tag image datasets from timelapse video, and your job is to **make it reproducible, scalable, and observable**.

This pipeline will run in the cloud for multiple client sites, so reliability, modularity, and cost awareness are important.

Your Tasks

You'll start from a basic video → image → pre-tag pipeline and enhance it with robust MLOps practices.

1. Pipeline Containerisation

- Containerise the dataset generation pipeline (e.g., using Docker).
- The container should accept as inputs:
    - Path to input video
    - Output directory
    - Detection model config (or use a default)

- Provide instructions or a script to run the container.

2. Reproducibility & Automation

- Implement a CI/CD-style script (e.g., using GitHub Actions or shell + Makefile) to:
    - Set up the environment
    - Run the pipeline end-to-end
    - Validate that outputs are correctly generated
- Add a simple test that checks that the output images and COCO file exist and are valid.
- Include **unit tests** for key functions in the pipeline (e.g., frame extraction, image saving, detection parsing). You may use a testing framework of your choice (e.g., pytest, unittest).

3. Data Observability & Reporting

- Add simple logging and metrics to track:
    - Number of images extracted
    - Number of detections per frame
    - Frame drop ratio (bad or duplicate frames)
- Output a short HTML or Markdown report summarising:
    - Dataset stats (image count, detection count, class distribution)
    - Time breakdown of each pipeline stage
- Bonus: If you're comfortable, demonstrate integration with a monitoring tool like W&B, Prometheus, or basic CSV logging.

4. Optional Enhancements (choose 1–2)

You are not required to do all of these, but pick any that showcase your strengths:

- Add frame deduplication using perceptual hashing or frame similarity
- Implement basic input validation (e.g., video format checks)
- Write IaC (Terraform/Cloudformation) to provision minimal cloud infra for the task
- Clean and pre-tag images using an off-the-shelf object detection model (YOLOv5/YOLOv8 or from Hugging Face)

Inputs

- Sample input .mp4 timelapse video (~2 mins) (Link here)
- A basic video_pipeline.ipynb notebook (we will provide)

Submission Requirements

Please provide the following in your submission:

1. **Code**
    - A zip folder or GitHub repo with:
        - Dockerfile
        - Pipeline scripts
        - CI/CD or automation scripts
        - README with clear instructions
2. **Report**
    - Markdown or HTML file with:
        - Summary of pipeline functionality
        - Dataset statistics and observability metrics
        - Description of improvements you'd make in production
3. **5–10 min Video**
    - Overview of your pipeline and design decisions
    - Rationale for your tooling choices
    - Suggestions for improvements or scaling to many sites
