📹 Hiring Assignment: Video Analysis Agent

🎯 Objective

Build an **Analysis Agent** that evaluates whether a Hercules test run was executed as planned. It does this by comparing:

- The **agent's planning log (thoughts/steps)**
- The **video evidence** of the run
- The **final test output**

Your goal is to detect if the steps the agent *claimed* to take are actually *visible in the video*. If any claimed action was skipped, altered, or not visibly performed — your agent should flag the deviation.

📥 Inputs

You'll be provided with the following Hercules test artifacts:

- 🎥 One or more **video recordings** of the test run.
- 🧠 A **planning log** representing the agent's internal steps or reasoning.
- 📄 The **final test output** as generated by Hercules.

⚙️ Processing Steps

1. **Parse the Planning Log**

   Extract the intended step-by-step actions the agent was supposed to take.

2. **Inspect the Video(s)**

   Determine whether each claimed action is visibly executed in the video(s).

Multiple videos may need to be coordinated for full coverage.

3. **Cross-check with Final Output**

   Use the final test output as a reference to validate consistency and outcome alignment.

🧾 Expected Output

Produce a **deviation report** that, for each claimed step, indicates:

Step Description	Result	Notes
Click “Login”	✅ Observed	-
Enter Password	❌ Deviation	Step skipped in video from 00:21
Submit Form	✅ Observed	

If everything matches:

➡️ **“No deviations detected.”**

🛠️ Requirements

- **Language/Framework**: Your choice. Use Python, TypeScript, etc. Use orchestration frameworks like **AutoGen**, **LangChain**, **Haystack**, or **your own tooling**.
- **Accuracy > Performance**: Focus on getting correct matches and clear reports rather than hyper-optimization.

- **Transparency**: Explain your decisions in README and video.

📁 Deliverables

1. **GitHub Repository**
    - Source code for the analysis agent
    - Brief, clear README including:
        - How to run the agent
        - Where output is saved
        - Sample inputs and expected outputs
2. **Video Walkthrough** (Screen Recording)
    - Explain your approach: input ingestion → step comparison → report generation
    - Show a live run on sample Hercules artifacts
    - Display of the final deviation report

🧪 Evaluation Criteria

Area	What We're Looking For
✅ Correctness	Accurately detects observed vs. deviated actions
🧠 Thought Process	Rational explanation of architecture & methodology

🧼 Code Quality	Clean structure, readable, modular design
📚 Documentation	Easy-to-follow README and instructions
🧰 Usability	Plug-and-play: drop files in → get a report out

🔗 Reference

- 🔍 Hercules GitHub: https://github.com/test-zeus-ai/testzeus-hercules