Project idea breakdown

1. Admin Mode 1 (Region of Interest Definition): - Live video feed display - Ability to draw and label multiple regions of interest (ROIs) on the video - Save ROI coordinates and labels to a text file

2. Admin Mode 2 (Process Definition): - Interface to define the correct process sequence - Input the order of ROI interactions (e.g., "Hand moves from ROI 1 to ROI 3, then to ROI 2") - Save this process definition to a text file

3. User Mode: - Live video feed with overlaid ROIs - Real-time hand tracking - Check if hands are moving between ROIs in the correct sequence - Provide feedback on process correctness

1. Technologies:
   * Python as the main language
   * OpenCV for video processing
   * MediaPipe for hand tracking
   * PyQt5 for the user interface
   * Simple text files for data storage
2. Key Components:

a) ROI Detector:

* + Use OpenCV to allow drawing rectangles on the video feed
  + Save and load ROI definitions

b) Hand Tracker:

* + Use MediaPipe Hands to track hand landmarks
  + Determine which ROI(s) the hands are in

c) Process Validator:

* + Compare the sequence of ROI interactions with the defined correct process
  + Provide real-time feedback

d) User Interface:

* + Admin ROI Window: For drawing and labeling ROIs
  + Admin Process Window: For defining correct process sequences
  + User Window: For live process validation

1. Data Storage:
   * roi\_definitions.txt: Store ROI coordinates and labels
   * process\_definitions.txt: Store correct process sequences
2. Development Steps:

a) Implement ROI Definition:

* + Create UI for drawing ROIs on video feed
  + Develop functions to save/load ROI definitions

b) Implement Process Definition:

* + Create UI for defining correct process sequences
  + Develop functions to save/load process definitions

c) Implement Hand Tracking:

* + Integrate MediaPipe Hands
  + Develop function to determine which ROI a hand is in

d) Implement Process Validation:

* + Create logic to compare hand movements with defined process
  + Develop real-time feedback mechanism

e) Create User Mode:

* + Combine all components for live process validation
  + Implement visual feedback for correct/incorrect actions

.

Here’s an updated \*\*User Mode breakdown\*\* incorporating the specifics:

---

### \*\*User Mode Breakdown\*\*

1. \*\*Interface Setup\*\*:

- When running `user.py`, the interface opens with a live video feed displaying overlaid regions of interest (ROIs).

- The user can see their hand movements relative to the defined ROIs, enabling real-time visual feedback.

2. \*\*Hand Labeling and Tracking\*\*:

- Hands are labeled as \*\*Hand 1\*\* and \*\*Hand 2\*\* based on entry order into the frame.

- Hands retain labels and are required to complete any initiated process (e.g., Hand 1 must finish the process it starts).

- MediaPipe’s detection may falter when hands are perpendicular to the camera (90 degrees), as all 21 points align, potentially leading to mislabeling.

- To address this, a \*\*dequeue\*\* structure holds recent frames and applies a \*\*weighting/voting system\*\* to determine the correct hand label, ensuring consistency.

3. \*\*Process Sequence Validation\*\*:

- Each cycle follows a set of defined processes (e.g., 4 in `process\_definitions.txt`), validated in sequence as per the hand initiating the process.

- Both hands are tracked simultaneously, with each hand moving through the ROIs in the correct sequence.

4. \*\*Cycle Completion and Feedback\*\*:

- When all processes in a cycle are correctly completed, the message \*\*"Task executed successfully!"\*\* displays.

- After a 2-second pause following each successful cycle, tracking for the next cycle resumes automatically.

- \*\*Cycle counts\*\* are tracked and displayed as follows:

- \*\*Total cycles\*\*

- \*\*Correct cycles\*\*

- \*\*Incorrect cycles\*\*

5. \*\*Error Detection and Restart\*\*:

- If any process is executed incorrectly, tracking stops, and \*\*"Wrong process executed"\*\* displays.

- The cycle restarts from the beginning (Process 1) upon pressing the \*\*Restart Cycle\*\* button on the UI.

- The count updates accordingly.

6. \*\*Continuous Cycle Tracking\*\*:

- The system continuously tracks each cycle without manual initiation, restarting after each completed or restarted cycle.

- Counts and real-time feedback update dynamically in the UI, enhancing user clarity.

I'll analyze the User Mode breakdown and create an organized implementation plan. Here's the step-by-step approach:

1. \*\*Hand Detection and Basic Tracking Setup\*\*

- Integrate MediaPipe Hands library

- Set up basic hand landmark detection

- Display detected hands on the video feed

2. \*\*Hand Labeling System\*\*

- Implement Hand 1 and Hand 2 labeling based on entry order

- Create dequeue structure for recent frames

- Develop weighting/voting system for consistent hand labeling

- Handle perpendicular hand cases (when 21 points align)

3. \*\*Process Sequence Management\*\*

- Load and parse process\_definitions.txt

- Create a state machine to track current process for each hand

- Implement ROI intersection detection (determine when a hand is in a ROI)

- Track sequence progression for each hand independently

4. \*\*Cycle Management\*\*

- Implement cycle start/end detection

- Add 2-second pause between successful cycles

- Create automatic cycle restart mechanism

- Enhance cycle counter system (total/correct/incorrect)

5. \*\*Validation and Error Handling\*\*

- Implement process sequence validation

- Add error detection for incorrect sequences

- Create validation feedback system

- Handle cycle interruptions and restarts

6. \*\*UI Enhancements\*\*

- Add real-time process status indicators

- Display current hand positions and labels

- Show active process step for each hand

- Enhance status messages for different scenarios

- Add visual feedback for correct/incorrect movements

7. \*\*Status and Feedback System\*\*

- Implement "Task executed successfully!" message

- Add "Wrong process executed" warning

- Create visual indicators for cycle progress

- Display current step in the process sequence

Let me know which step you'd like me to implement first, and I'll provide the code for that component. I recommend starting with Step 1 (Hand Detection and Basic Tracking) as it's the foundation for all other functionality.

Would you like me to proceed with implementing the hand detection system first? Or would you prefer to start with a different component?