



# AI IN THE STEEL INDUSTRY:

## *AI-POWERED IRON BAR INSPECTION SYSTEM*

### TEAM JUSTEEL

DONGIK MIN  
GURUPREET SINGH  
ABEYA JALE BIKILA  
JOSEPH  
BAKHODIR

# OUR TEAM JUSTEEL



Dongik Min



Gurupreet  
Singh



Bakhodir



Jale Abeya



Jospheh

# Table of contents

- 1. THE PRODUCTION PROCESS**
- 2. PROBLEM DEFINITION**
- 3. SOLUTION**
- 4. ARCHITECTURE**
- 5. DESIGN (EXPLANATION)**
- 6. DEMO**
- 7. RESULTS**
- 8. WHAT'S NEXT..?**



# IRON BAR PRODUCTION PROCESS



**our project will be deployed here**



# PROBLEM DEFINITION

- **Human error:** Potential for mistakes in manual inspections, such as miscounting bars or misreading sticker information.
- **Inefficiency:** Manual inspection is time-consuming and labor-intensive, leading to slower processing times and higher operational costs.

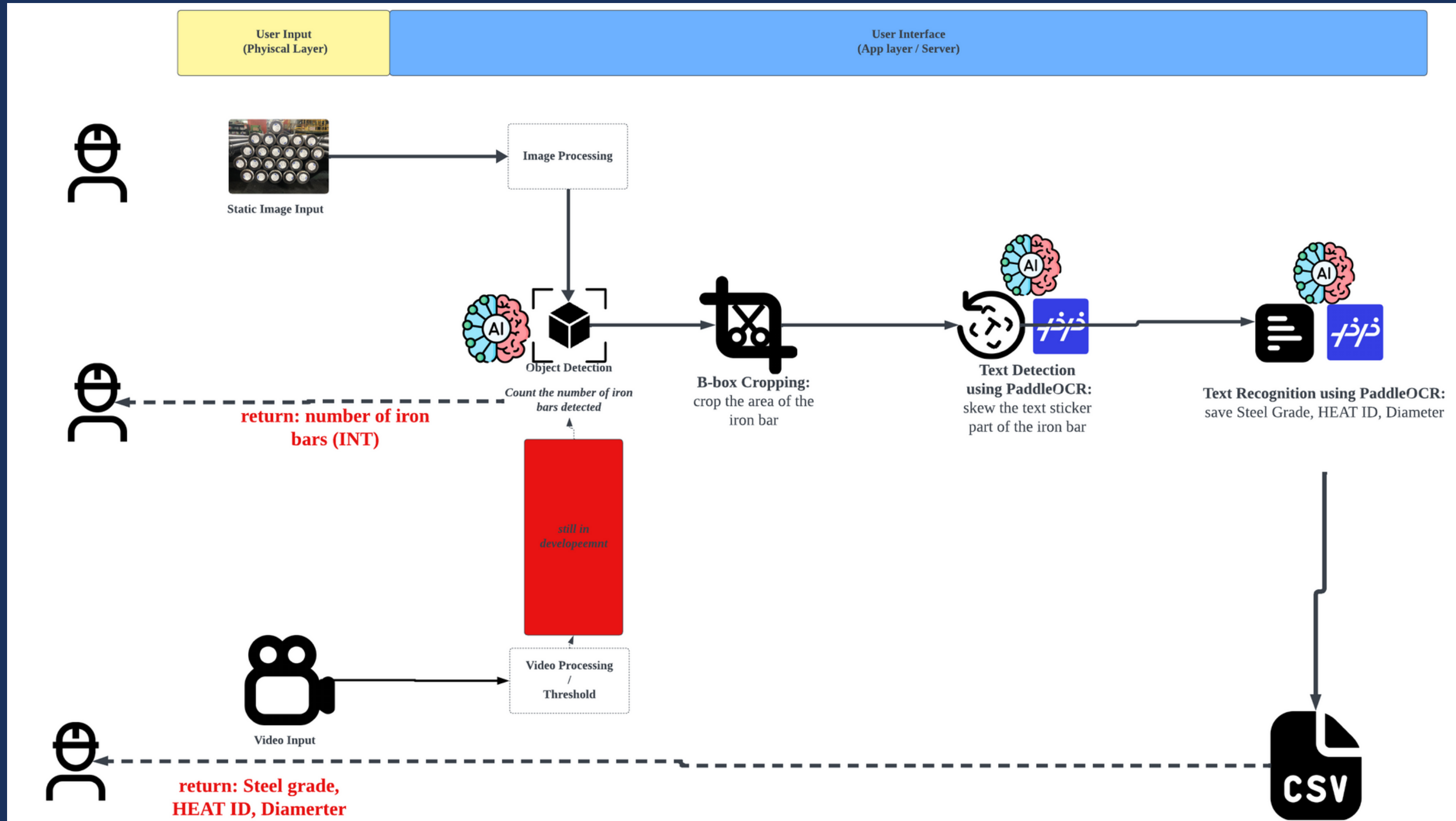


# SOLUTION

- **Automated Object Detection:** AI algorithms accurately detect and count the number of iron bars in both static images and video feeds.
- **Text Detection and Recognition:** OCR extracts and recognizes text labels such as steel grade, HEAT ID, and diameter, ensuring accurate documentation.
- **Efficient Data Output:** Recognized data is automatically saved into CSV files, facilitating easy access and management.
- **Integrated Video Processing:** The system processes video inputs to provide continuous and real-time monitoring, currently under development to enhance system capabilities.



# ARCHITECTURE

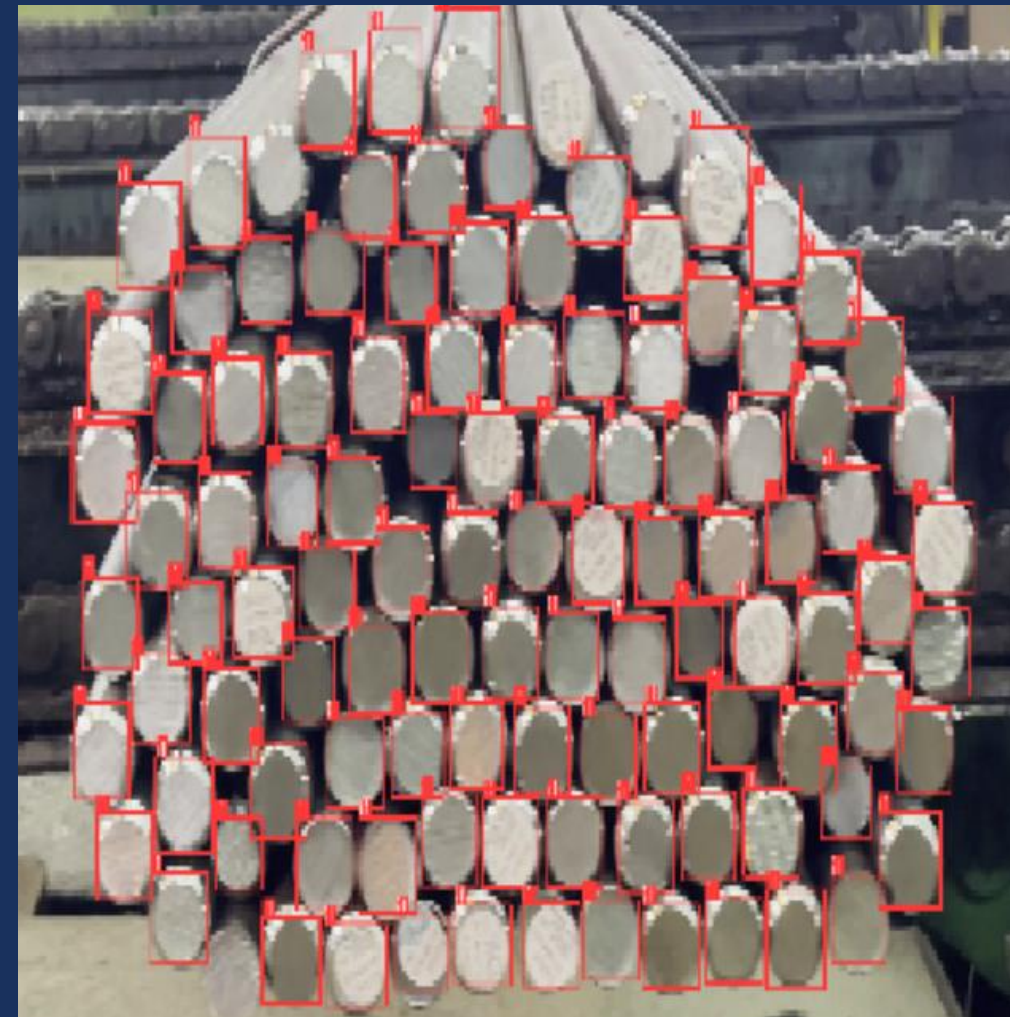
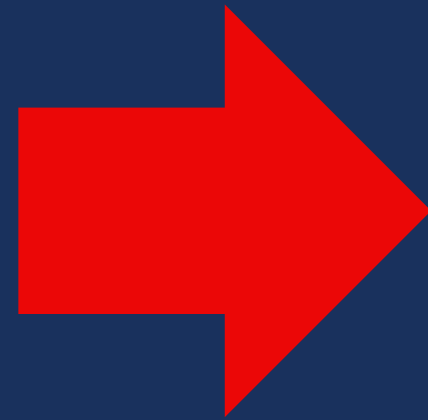




# ARCHITECTURE EXPLANATION

## OBJECT DETECTION – STATIC IMAGE

*n number of iron bar... (count the boundary box)*



52.6 ms per one image



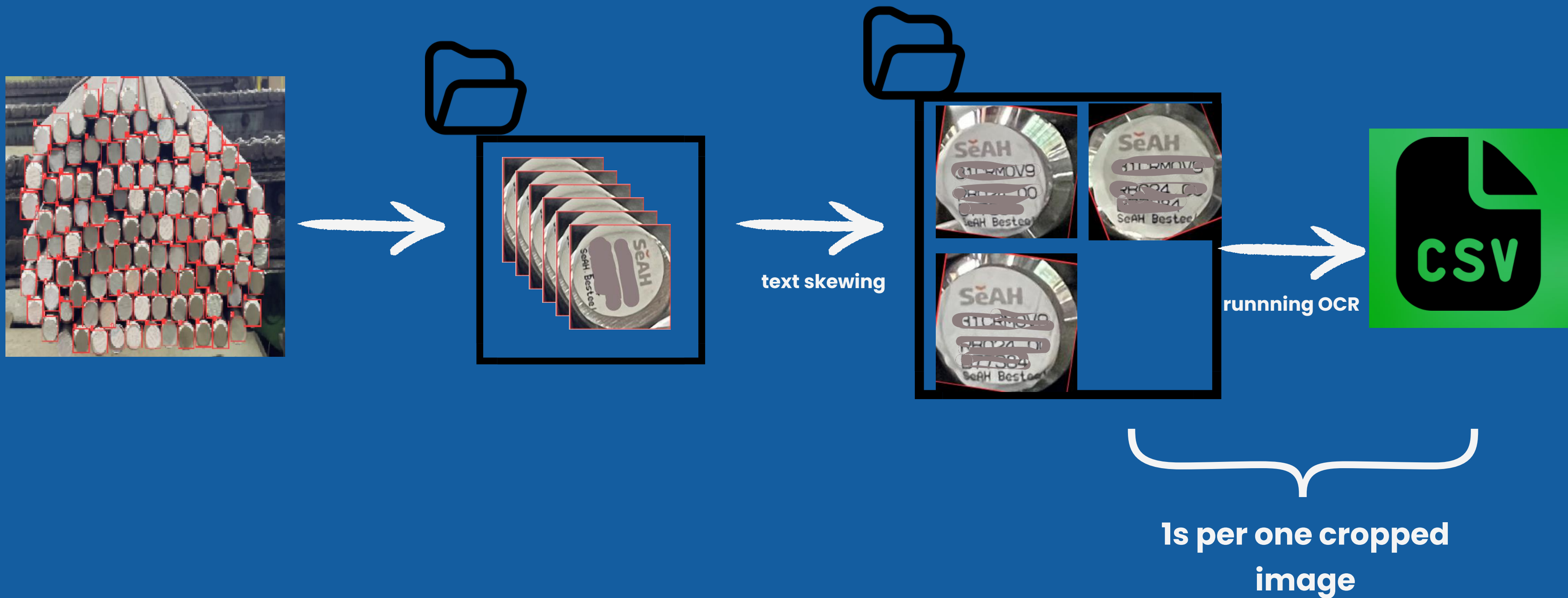
# ARCHITECTURE EXPLANATION

*object detection - real time video input (in development)*



# ARCHITECTURE EXPLANATION

## PREPROCESSING AND TEXT STICKER RECOGNITION





**DEMO  
VIDEO**

FINAL\_YOLOV8

- ultralalytics
  - \_\_pycache\_\_
  - .github
  - build
  - datasets
    - input\_image1.jpg
    - vid\_test1.mp4
    - vid\_test2.mp4
  - docker
  - docs
  - examples
  - models
  - results
  - tests
  - ultralalytics
  - ultralalytics.egg-info
- .gitignore
- .pre-commit-config.yaml
- CITATION.cff
- CONTRIBUTING.md
- detections.csv
- LICENSE
- main\_image\_copy.py
- main\_image.py
- main\_video.py
- mkdocs.yml
- output.csv
- pyproject.toml
- README.md
- README.zh-CN.md
- rotate\_and\_ocr.py
- with\_paddleocr\_test\_copy.py
- with\_paddleocr\_test.py

계묘

타일자본

```

ultralalytics > with_paddleocr_test.py > calculate_angle
75 # Get the image file name
76 image_file_name = os.path.basename(rotated_image_path).split('.')[0]
77
78 # Check if the CSV file exists
79 if not os.path.exists(csv_file_path):
80     # Create a new CSV file with header
81     with open(csv_file_path, 'w', newline='') as csvfile:
82         fieldnames = ['Image_file_name', 'Bar No.', 'HEATID', 'Type', 'Steel_grade']
83         writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
84         writer.writeheader()
85
86 # Read the existing data from the CSV file
87 existing_data = []
88 with open(csv_file_path, 'r') as csvfile:
89     reader = csv.DictReader(csvfile)
90     for row in reader:
91         existing_data.append(row)
92
93 # Find the appropriate columns for the detected texts based on similarity
94 row_count = len(existing_data) + 1
95
96 for i, text in enumerate(filtered_texts[:3], start=row_count):
97     row_data = {'Image_file_name': image_file_name, 'Bar No.': str(i)}
98
99     for column in ['HEATID', 'Type', 'Steel_grade']:
100         best_match_text = None
101         best_match_ratio = 0
102
103         for existing_text in [row[column] for row in existing_data]:
104             match_ratio = difflib.SequenceMatcher(None, text, existing_text).ratio()
105             if match_ratio > best_match_ratio:
106                 best_match_text = existing_text
107                 best_match_ratio = match_ratio
108
109         row_data[column] = best_match_text if best_match_text else text

```

문제 중지 디버그 콘솔 터미널 모드 실행 모니터

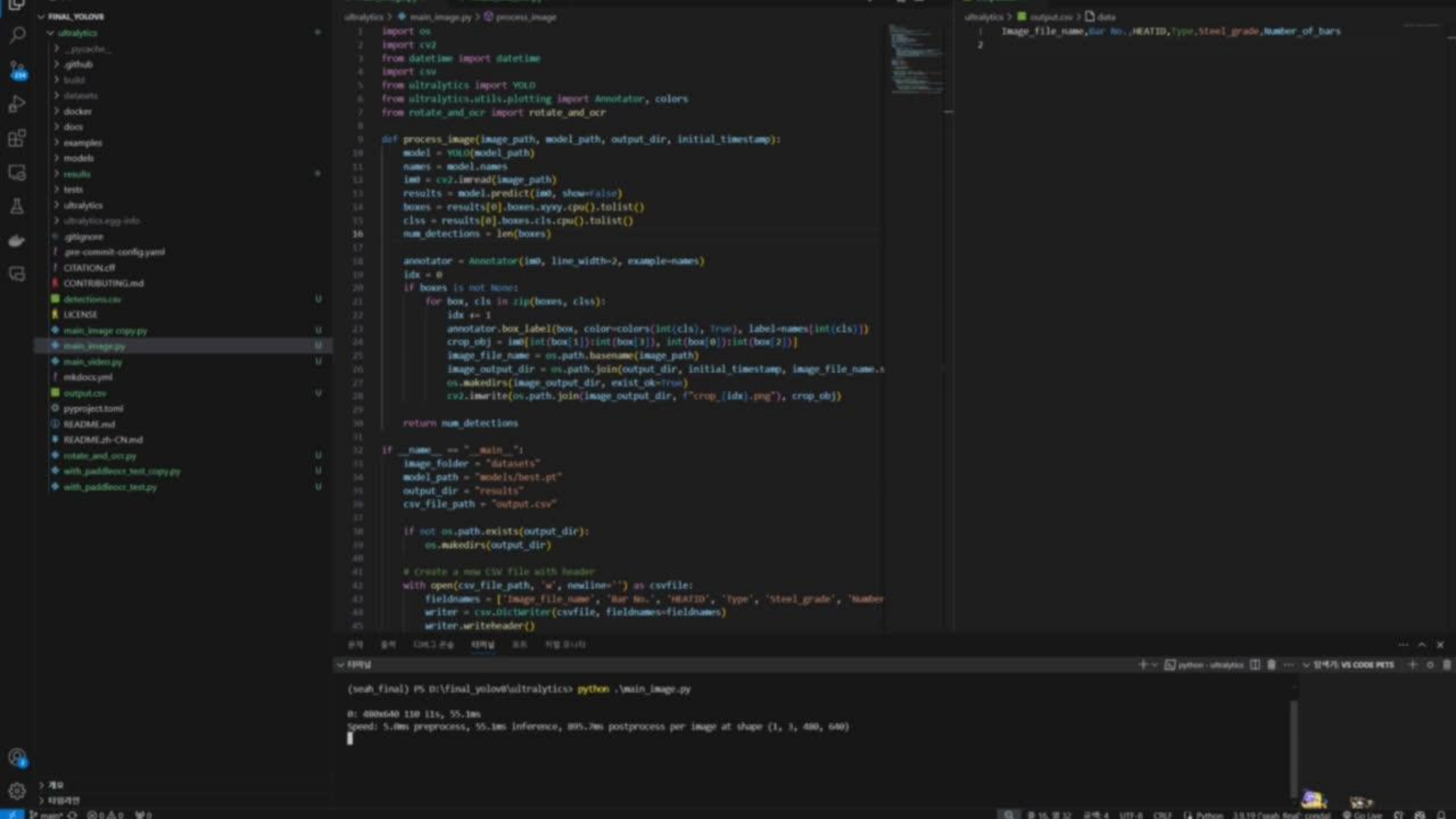
터미널

(seah\_final) PS D:\final\_yolov8\ultralalytics> []

검색어: VS CODE PETS

종 12.월 57 공백 4 UTF-8 CRLF Python 3.9.19 (seah\_final) conda Go Live





```
ultralalytics > main_image.py > process_image
1 import os
2 import cv2
3 from datetime import datetime
4 import csv
5 from ultralytics import YOLO
6 from ultralytics.utils.plotting import Annotator, colors
7 from rotate_and_ocr import rotate_and_ocr
8
9 def process_image(image_path, model_path, output_dir, initial_timestamp):
10     model = YOLO(model_path)
11     names = model.names
12     img = cv2.imread(image_path)
13     results = model.predict(img, show=False)
14     boxes = results[0].boxes.xyxy.cpu().tolist()
15     cls = results[0].boxes.cls.cpu().tolist()
16     num_detections = len(boxes)
17
18     annotator = Annotator(img, line_width=2, example_names=names)
19     idx = 0
20     if boxes is not None:
21         for box, cls in zip(boxes, cls):
22             idx += 1
23             annotator.box_label(box, color=colors(int(cls), True), label=names[int(cls)])
24             crop_obj = img[int(box[1]):int(box[3]), int(box[0]):int(box[2])]
25             image_file_name = os.path.basename(image_path)
26             image_output_dir = os.path.join(output_dir, initial_timestamp, image_file_name)
27             os.makedirs(image_output_dir, exist_ok=True)
28             cv2.imwrite(os.path.join(image_output_dir, f"crop_{idx}.png"), crop_obj)
29
30     return num_detections
31
32 if __name__ == "__main__":
33     image_folder = "datasets"
34     model_path = "models/hest.pt"
35     output_dir = "results"
36     csv_file_path = "output.csv"
37
38     if not os.path.exists(output_dir):
39         os.makedirs(output_dir)
40
41     # Create a new CSV file with header
42     with open(csv_file_path, 'w', newline='') as csvfile:
43         fieldnames = ["Image_file_name", "Bar No.", "HEATID", "type", "Steel_grade", "Number_of_bars"]
44         writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
45         writer.writeheader()
```

```
ultralalytics > output.csv > data
1 Image_file_name, Bar No., HEATID, type, Steel_grade, Number_of_bars
2
```

```
python .\main_image.py
0: 480x640 110 11s, 55.1ms
Speed: 5.0ms preprocess, 55.1ms inference, 895.7ms postprocess per image at shape (1, 3, 480, 640)
```

# PROJECT RESULTS



# Result 1

Text: ██████████

Confidence: 0.38

Bounding box: [[41.05025253169417, 160.05025253169416], [303.56424132506595, 122.56879950923484], [309.9497474683058, 179.94974746830584], [47.43575867493408, 217.43120049076515]]

Text: ██████████

Confidence: 0.66

Bounding box: [[55.20891722009885, 223.58050960622737], [226.3093821920085, 201.96821894669722], [230.79108277990116, 259.41949039377266], [59.69061780799148, 282.0317810533028]]

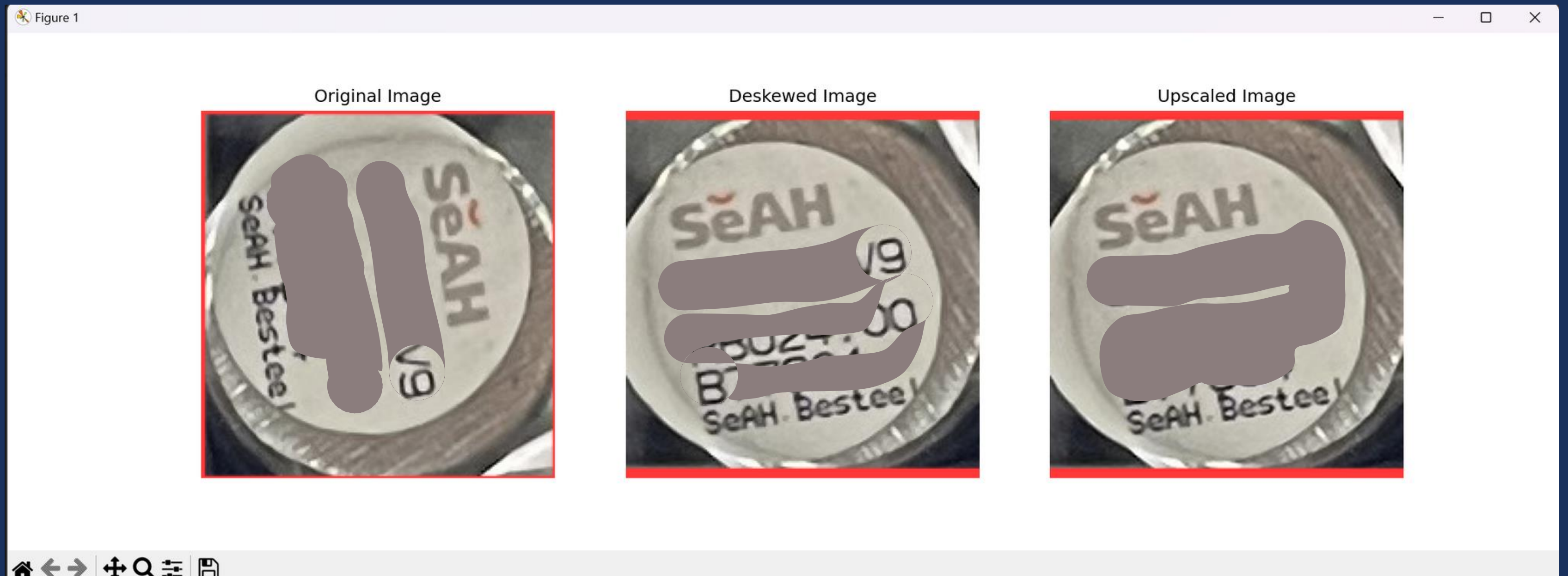
Text: B ██████████

Confidence: 0.91

Bounding box: [[61.93133264396633, 268.30386570155287], [262.3495412261482, 236.053251585655], [269.0686673560337, 293.69613429844713], [68.65045877385182, 325.946748414345]]

The result shown here is the picture of the detected texts with accuracy levels.

# Result 2



The above picture shows ;

- the original image of one steel bar
- cropped imaged and skewed image





# Result 3

```
Image_file_name,Bar No.,HEATID,Type,Steel_grade,Number_of_bars
rotated_20240530_171358,28,
rotated_20240530_171358,29,
rotated_20240530_171409,30,
rotated_20240530_171409,31,
rotated_20240530_171409,32,
rotated_20240530_171413,33,
rotated_20240530_171413,34,
rotated_20240530_171413,35,
rotated_20240530_171422,36,
rotated_20240530_171429,37,
rotated_20240530_171433,38,
rotated_20240530_171433,39,
rotated_20240530_171433,40,
rotated_20240530_171441,41,
```

Here the result shown is a blurred image of csv output.

# WHAT'S NEXT?

- Seamless Video Integration
  - Code 
  - In-factory camera 
- Solve time complexity issue with OCR
  - Highly dependent on the performance of the GPU
  - (e.g. GPU RAM 8GB) Based on the PaddleOCR benchmark 1s/image

**ALL CROPPED STICKER**



**ONE CROPPED STICKER  
PER SINGLE BUNDLE**



**THANKS**

For Your Attention