16/06/2025

Training Day-84

Day 6: GUI Planning and PyQt Design Draft

Started GUI development by outlining the layout and required components. Planned for a dark-themed dashboard interface with tabs and navigation on the left. Identified key modules including image loader, detection trigger, export button, webcam integration, and batch processing. Created wireframes and began coding the first layout with QLabel and QPushButton elements. Connected signal-slot mechanisms to enable responsive actions. Ensured GUI was modular and ready to integrate backend inference. Design decisions were made to keep the interface clean, modern, and user-friendly.

Objective:

GUI Planning and PyQt Design Draft

Outcome:

Successfully completed implementation and verification of the day's objectives using Python, PyQt5, YOLOv8, and supporting libraries.

17/06/2025

Training Day-85

Day 7: Integrating YOLOv8 with PyQt GUI

Successfully connected the YOLOv8 trained model to the PyQt GUI using ultralytics Python API. Enabled file selection and passed images to the model for detection. Captured inference results and visualized bounding boxes in QLabel using OpenCV and QImage conversion. Added error handling for missing models and invalid images. Detection results were summarized in a QTextEdit widget. The detection flow from file → model → result → display was established completely on this day, forming the backbone of the application.

Objective:

Integrating YOLOv8 with PyQt GUI

Outcome:

Successfully completed implementation and verification of the day's objectives using Python, PyQt5, YOLOv8, and supporting libraries.

18/06/2025

Training Day-86

Day 8: Implementing Batch Processing and Export Features

Developed functionality to process entire image folders. Looping over files, inference was executed and results were stored in a Pandas DataFrame. Each image’s detected and missing components were tracked. The report was exported to Excel using `to_excel()` function, supporting easy analysis and review. Additionally, reset functionality was added to clear the GUI state. The day also included testing edge cases such as empty folders and unreadable files. The batch feature greatly increased the utility and scalability of the tool.

Objective:

Implementing Batch Processing and Export Features

Outcome:

Successfully completed implementation and verification of the day's objectives using Python, PyQt5, YOLOv8, and supporting libraries.

19/06/2025

Training Day-86

Day 9: Advanced Features: QR Code and Summary Charts

QR code generation was added for each image ID using the `qrcode` library. QR images were saved and linked to detection IDs. Also implemented summary graphs using matplotlib. Pie charts and stacked bar charts displayed component detection rates and quality trends across datasets. The visualizations enhanced the interpretability of results and added analytical depth. These advanced features made the application presentation-ready and useful for production quality assessments.

Objective:

Advanced Features: QR Code and Summary Charts

Outcome:

Successfully completed implementation and verification of the day's objectives using Python, PyQt5, YOLOv8, and supporting libraries.

20/06/2025

Training Day-87

Day 10: Testing, Finalizing GUI, and Report Export

Performed end-to-end testing including GUI behavior, error handling, and batch operation results. Verified layout responsiveness, detection consistency, and export accuracy. Final touches like improving labels, fixing typos, aligning buttons, and updating icons were completed. Confirmed that all functionality including image detection, webcam inference, and QR generation worked smoothly. Exported final test results and created sample reports for submission. Prepared documentation and confirmed readiness for evaluation.

Objective:

Testing, Finalizing GUI, and Report Export

Outcome:

Successfully completed implementation and verification of the day's objectives using Python, PyQt5, YOLOv8, and supporting libraries.