DECEMBER 2025

CIRCUITGUARD - PCB DEFECT DETECTION SYSTEM

AN AI-POWERED PCB DEFECT IDENTIFICATION & CLASSIFICATION PLATFORM

Prepared by: Prashant Yadav
Infosys Internship Batch 6

Mentor:
Mr. Farman

INTRODUCTION

Printed Circuit Boards (PCBs) are the core of all electronic systems, and even minor defects can result in device malfunction, short circuits, or long-term reliability issues. Traditional manual inspection is time-consuming and error-prone.

This project presents CircuitGuard, an AI-based PCB defect detection system that utilizes deep learning and image processing to automatically detect and classify PCB defects with high precision.

The system integrates a trained YOLO model with a Streamlit-based interface, enabling fast, accurate, and user-friendly defect evaluation.

PROBLEM STATEMENT

Manual PCB inspection is inefficient, inconsistent, and unsuitable for high-volume manufacturing.

There is a need for an automated system that can:

- Detect defects with high accuracy
- Classify defect types
- Visualize bounding boxes and confidence scores
- Provide real-time analysis through a web interface

This project aims to eliminate manual errors and provide a scalable automated defect recognition solution.

PROJECT OBJECTIVES

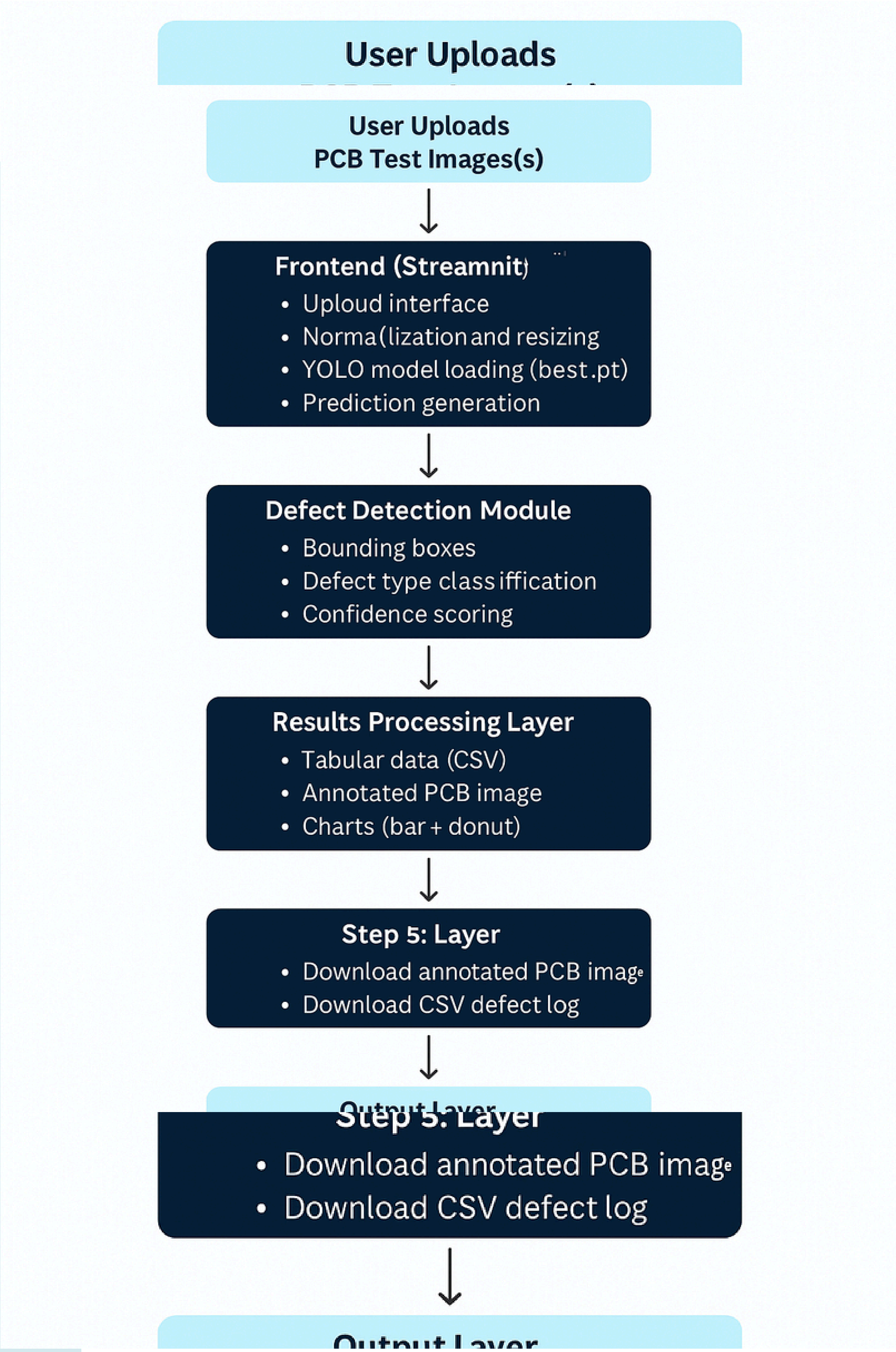
The next step is to decide what metrics matter to you. Below are some metrics I started with::

- Automate PCB defect detection using deep learning
- Identify and classify common defect types such as spur, missing hole, short, mouse bite, etc.
- Generate bounding box-based visual annotations
- Build an interactive UI for uploading and inspecting PCB images
- Display detailed analytics including defect counts and type distribution
- Enable export of annotated images and CSV logs

SYSTEM ARCHITECTURE

Flow:
Image Upload → Preprocessing → YOLO Model Inference → Defect Detection → Bounding Box Visualization → Results Dashboard → Export

Components:
- Frontend: Streamlit UI
- Backend: Python inference pipeline
- Model: YOLO-based defect detector (trained using best.pt)
- Output: Annotated images, defect tables, charts, CSV logs


User Uploads
User Uploads
PCB Test Images(s)
Frontend (Streamnit)
Uploud interface
Norma(lization and resizing
YOLO model loading (best.pt)
Prediction generation
Defect Detection Module
Bounding boxes
Defect type classiffication
Confidence scoring
Results Processing Layer
Tabular data (CSV)
Annotated PCB image
Charts (bar + donut)
Step 5: Layer
Download annotated PCB image
Download CSV defect log
Download annotated PCB image
Download CSV defect log

METHODOLOGY WORKFLOW:

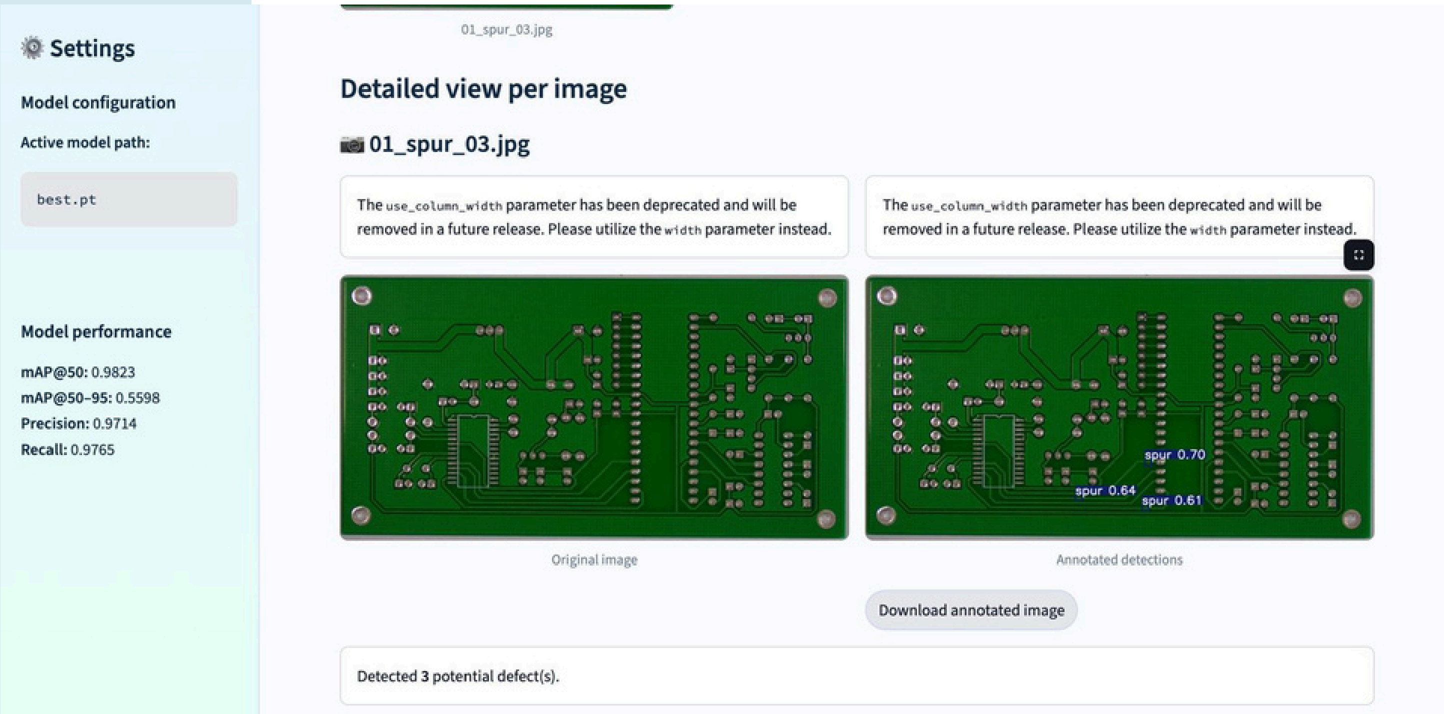
- Image Preprocessing
- Alignment, resizing, normalization
- Model Inference
- YOLO model generates predictions (x1, y1, x2, y2, defect type, confidence)
- Post-Processing
- Bounding boxes drawn on PCB image
- Result Visualization
- Tables, bar charts, donut charts
- Export Function
- Download annotated image + CSV defect log

Defect locations (bounding boxes in pixels):

	Defect type	Confidence	x1	y1	x2	y2
0	spur	0.7	1665.4	1099.5	1705.1	1139
1	spur	0.64	1253	1311.6	1296.2	1345.1
2	spur	0.61	1648.6	1370.4	1690.7	1403.9

Overall defect distribution across all uploaded images

MODULE 1 – DATASET PREPARATION & IMAGE PROCESSING



- DeepPCB dataset explored and cleaned
- Template and test images prepared
- Noise reduction applied
- Defects extracted using preprocessing + model inference

Outcome: Ready-to-use dataset with standardized format

(Reference: Project guidelines PDF)

MODULE 2 – DEFECT DETECTION & ROI EXTRACTION

Using the YOLO model, defect regions are automatically localized through bounding boxes. The model predicts each defect type based on learned visual features and outputs class labels such as "spur" along with high-precision bounding coordinates.

These detected regions of interest (ROIs) are extracted and stored in structured format, enabling further analysis or training refinement. The bounding box extraction ensures accurate marking of defect locations while maintaining consistency across multiple images.

This module is a crucial step in operationalizing the detection pipeline, transforming raw model predictions into actionable insights.

MODULE 3 - MODEL TRAINING

The model was trained using high-quality PCB defect samples with diverse augmentation techniques to improve generalization. The YOLO architecture was selected due to its strong performance on object detection tasks and ability to process images in real time.

The trained model achieved outstanding performance:

- *mAP@50 = 0.9823, indicating extremely high detection precision*
- *Precision = 0.9714, showcasing low false positives*
- *Recall = 0.9765, ensuring defects are rarely missed*
- *Model training stability was observed through consistent accuracy and loss curves, proving the robustness of the training pipeline.*

FRONTEND DEVELOPMENT (STREAMLIT UI)

Features included:

- ***Upload PCB images (PNG/JPG)***
- ***Display before/after images***
- ***Show detection table with defect type, confidence, coordinates***
- ***Bar chart of defect distribution***
- ***Donut chart of defect type share***
- ***Export annotated image & CSV***

The UI is clean, responsive, and matches all requirements from Module 5.

The frontend is designed to be clean, intuitive, and highly interactive. Users can upload PCB images directly from their device and instantly view detection outcomes. The interface displays original and annotated images side-by-side, offering clear visual representation of all detected defects.

Additionally, detailed tables summarize defect type, confidence value, and bounding box coordinates. Graphical visualizations such as bar charts and donut charts help users quickly analyze defect trends across uploaded images.

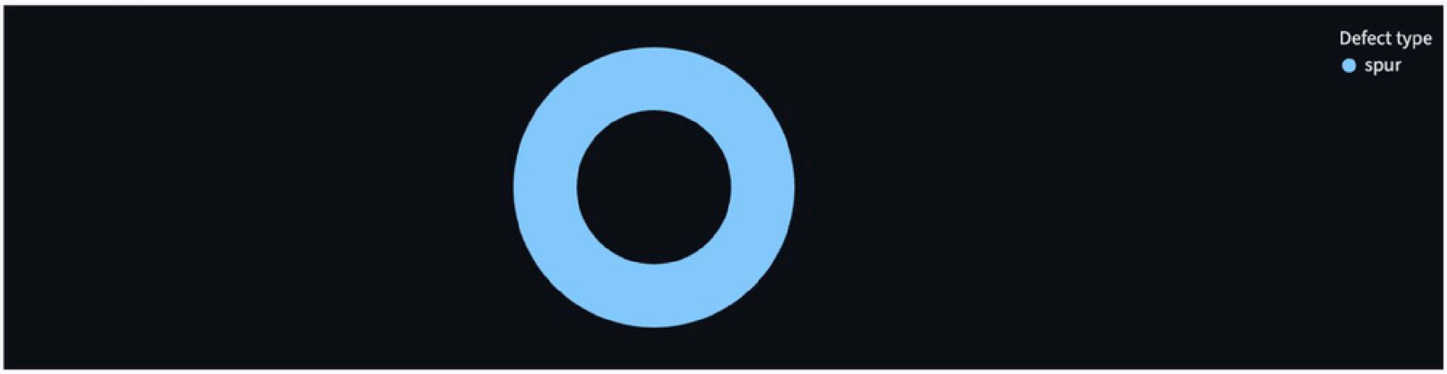
The frontend is responsive, fast, and fulfills all user experience requirements specified in the project guidelines.

BACKEND PIPELINE

The backend is responsible for executing the AI inference logic and preparing results for the frontend. Once the user uploads an image, the backend handles preprocessing, loads the YOLO model, and performs detection using efficient GPU-optimized techniques.

It extracts bounding boxes, converts results into pandas dataframes, generates annotated output images, and calculates statistical summaries. The backend pipeline is modular and optimized for minimal delay, ensuring smooth operation even when multiple images are uploaded.

This component forms the computational core of the CircuitGuard system and ensures reliable and repeatable defect detection.

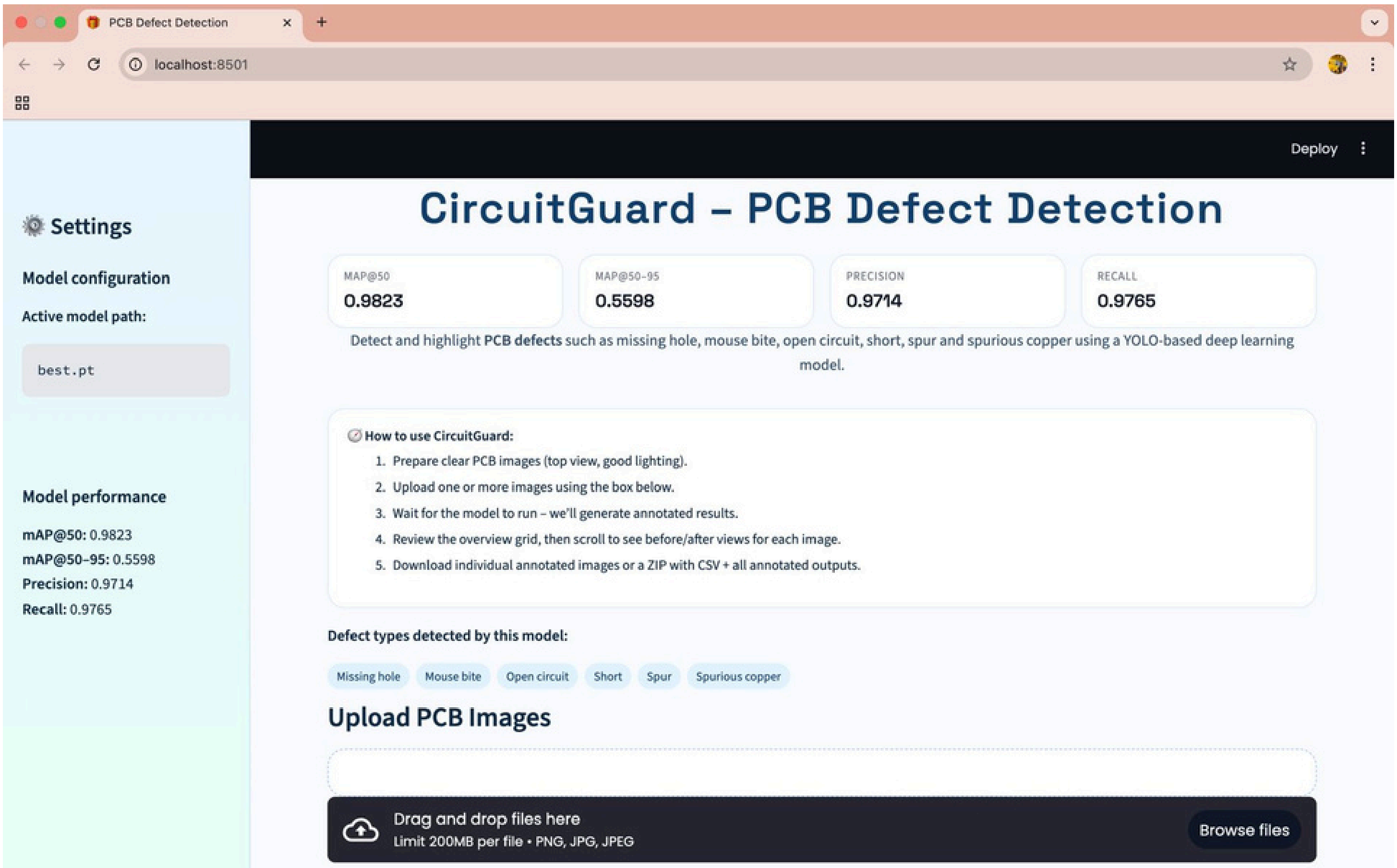


Export results

Finish defect detection

RESULTS & OUTPUTS

- The system successfully detected multiple spur defects with confidence scores ranging from 0.61 to 0.70. The annotated output images clearly marked the defect locations with labeled bounding boxes, providing easy interpretability.
- Charts revealed the distribution of defects across uploaded images, giving users an understanding of the frequency and severity of issues. The donut chart helped visualize defect share, confirming that all detected defects in the provided sample belonged to the "spur" category.
- The export functionality enabled seamless downloading of annotated images and structured CSV logs, demonstrating the system's practicality and completeness.



FUTURE SCOPE

- Extend training data to include additional PCB defect categories for broader coverage.
- Deploy the system as a cloud-based web service to support multi-user environments.
- Integrate batch processing to allow analysis of hundreds of PCB images at once.
- Develop a mobile-friendly interface for handheld inspection tools.
- Implement real-time inspection capability through video stream analysis.

CONCLUSION

CircuitGuard delivers a robust, efficient, and highly accurate PCB defect detection solution powered by deep learning. By integrating advanced AI models with a user-friendly interface, the system significantly enhances the speed and reliability of PCB inspection processes.

The automated workflow—from image upload to detection, visualization, and export—removes human inconsistencies and supports industrial-scale manufacturing environments. The strong performance metrics validate the effectiveness of the trained model, making CircuitGuard a powerful tool for quality assurance in electronics production.