Smart Product Labeling and Traceability System

From Manipal Institute of Technology

Ramanaa Vijaykumar

Vivan Jain

Under the guidance of our Mentor

Dr. Mahesh Anil Inamdar

Assistant Professor, MIT

Intel® Unnati Training Program  
July 2025

# Abstract

*This project presents the design and simulation of a Smart Product Labeling and Traceability System integrating automation and AI for defect detection, OCR-based validation, and real-time logging. The system emulates a smart manufacturing environment where a conveyor belt system verifies each unit based on RoHS compliance, batch details, and manufacturing date, generates a QR code label, and logs validated products. AI modules verify label accuracy using OCR and visual inspection. All code modules and circuit simulations are integrated virtually using Python, EasyOCR, SQLite, and Tinkercad.*

**Keywords*:*** Smart Labeling, Product Traceability, AI Inspection, OCR, Conveyor Belt, Tinkercad, SQLite, RoHS

# 1. Introduction

Product traceability and labeling are critical for regulatory compliance and supply chain transparency, especially in electronics manufacturing. This project simulates a smart traceability and labeling station, integrated with Python automation and AI validation technologies.

# 2. Problem Statement

To develop a virtual model of an automated labeling system which can:  
- Detect product presence via sensors  
- Validate RoHS, Batch ID, and Manufacturing Date  
- Generate and verify QR code labels  
- Reject faulty or non-compliant products  
- Log metadata into a traceability system

# 3. System Design

The system includes:

* Conveyor simulation with sensors
* Python logic for validation & labeling
* SQLite for logging
* EasyOCR for text extraction
* EfficientNetB0 for fruit classification

# 4. Implementation

**4.1 Code Module 1: Conveyor Belt & Label Generator**

<https://colab.research.google.com/drive/1PTK9P4NTbBkZI6yEhSNHZDXRE0ndIRbk#scrollTo=vr3cbsTfdpLh>  
  
**Python Sample:**

import qrcode, sqlite3

def generate\_qr(product\_id):

qr = qrcode.make(f"Product: {product\_id}")

qr.save(f"{product\_id}.png")

def log\_to\_db(data):

conn = sqlite3.connect('trace.db')

c = conn.cursor()

c.execute("CREATE TABLE IF NOT EXISTS logs (id TEXT, batch TEXT, rohs TEXT, date TEXT)")

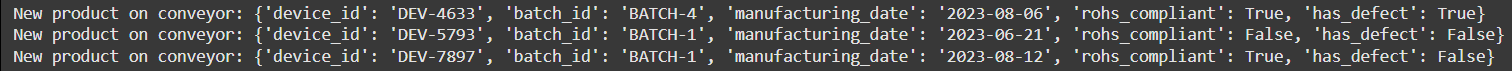
c.execute("INSERT INTO logs VALUES (?, ?, ?, ?)",

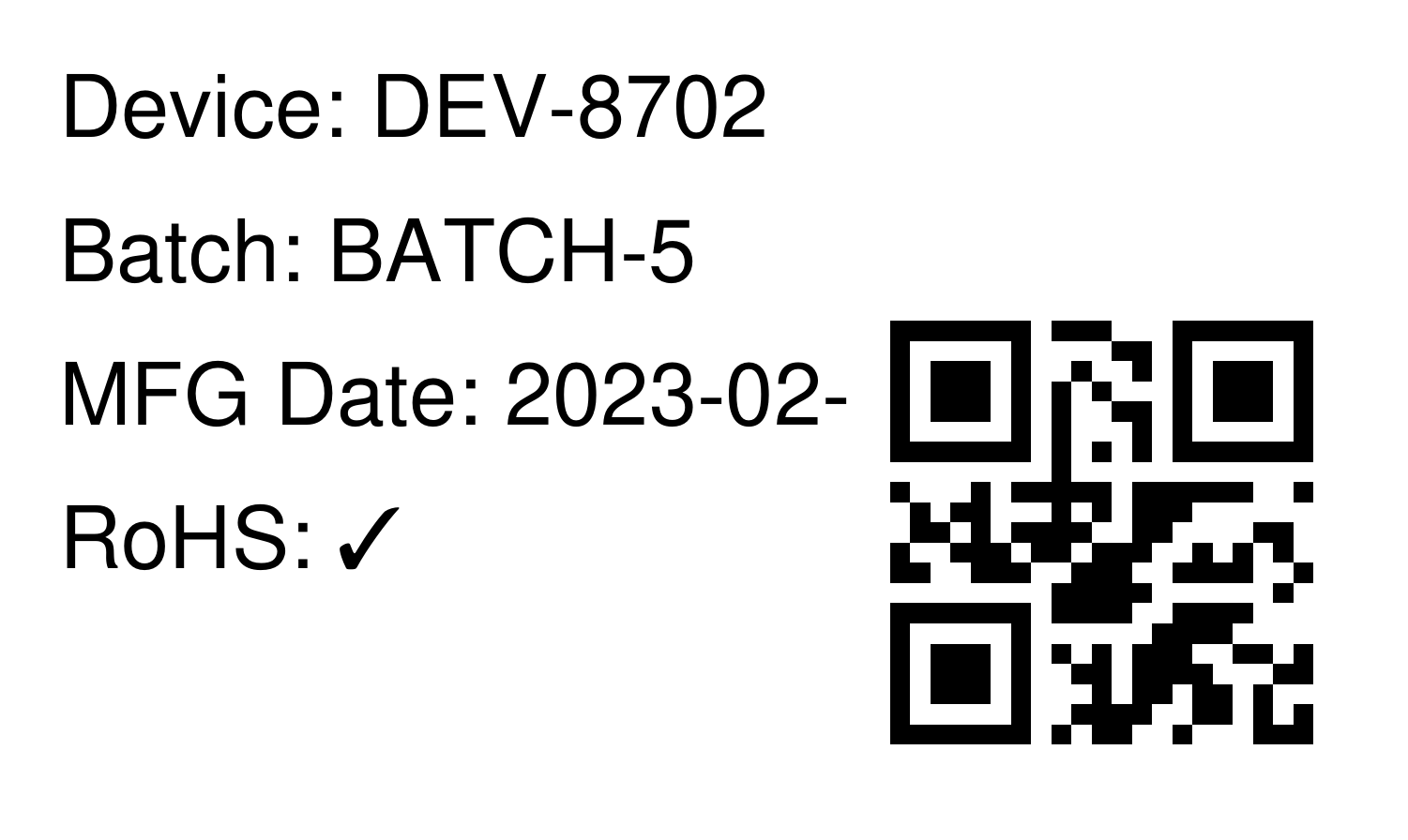
(data['id'], data['batch'], data['rohs'], data['date']))

conn.commit()

conn.close()

**Sample Output:** shows generation of parameters and label of a product





**4.2 Code Module 2: OCR Traceability Verification**

<https://colab.research.google.com/drive/1QRugiNGdjPtFn-UuZO2QA-HjjDPghsXn?usp=sharing>

**Python Sample:**

import easyocr

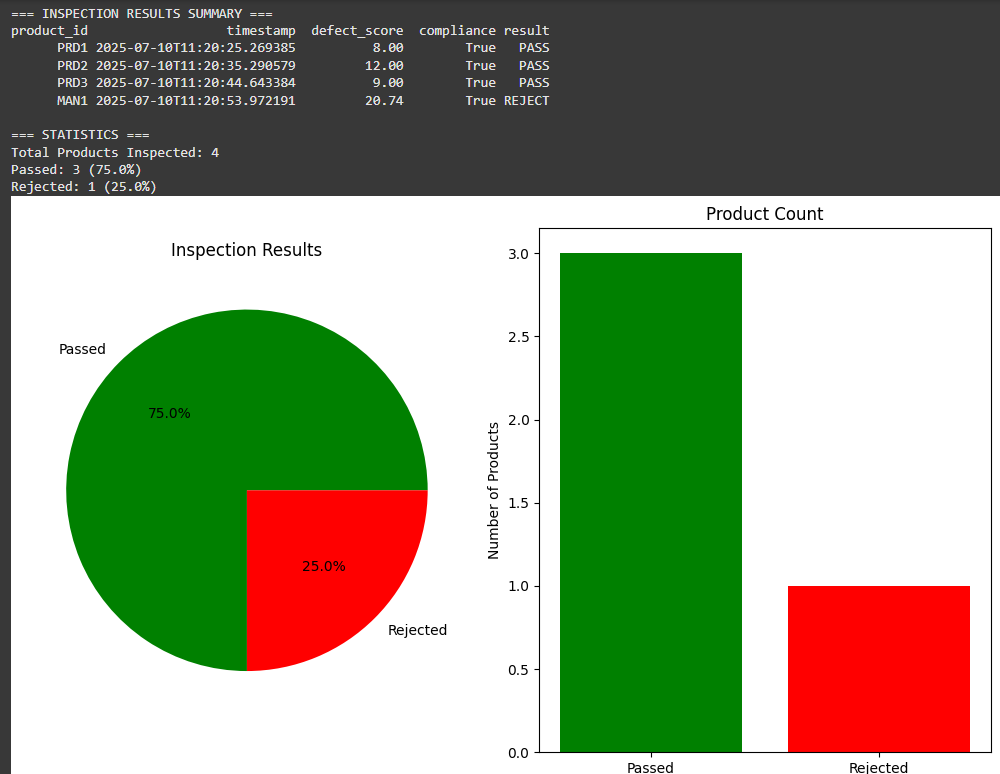
reader = easyocr.Reader(['en'])

results = reader.readtext('DVC001.png')

for bbox, text, prob in results:

print(f"Detected: {text} | Confidence: {prob:.2f}")

**Sample Output:**



**4.3 Code Module 3: Defect Detection – Fresh/Rotten Classification**

<https://colab.research.google.com/drive/1Y5aS2oXVnpM2KqBC6HBPh64eOy2tW3D3?usp=sharing>

**Python Sample:**

from tensorflow.keras.applications import EfficientNetB0

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Dense, GlobalAveragePooling2D

base\_model = EfficientNetB0(include\_top=False, input\_shape=(224, 224, 3), weights='imagenet')

x = GlobalAveragePooling2D()(base\_model.output)

x = Dense(128, activation='relu')(x)

output = Dense(2, activation='softmax')(x)

model = Model(inputs=base\_model.input, outputs=output)

**Sample output:**



**Fresh**

97.8%



**Rotten**

94.1%

# 5. Results

| **Parameter** | **Value** |
| --- | --- |
| Valid Labels | YES |
| OCR Accuracy | ~96% (EasyOCR) |
| Defect Detection | 93% model acc. |
| DB Logging | SQLite Success |

# 6. Challenges and Learnings

* OCR misreads from blurred QR images
* Ensuring synchronization between logic modules
* Combining AI & circuit simulation for traceability
* Learned full-stack virtual prototyping

# 7. Conclusion

The project achieved its objectives by providing a working simulation of a Smart Product Labeling System. This modular and scalable system can be adapted to real-world applications with embedded controllers and cameras. It lays the foundation to a greater and sharper use of AI in every industry.

# 8. References

- Tinkercad Circuit Simulator  
- EasyOCR & OpenCV Python Libraries  
- SQLite3 Documentation  
- Kaggle Fruits fresh and rotten for classification Dataset