COLLEGE CODE: 1128

COLLEGE NAME: T.J.S.ENGINEERING COLLEGE

DEPARTMENT: B.TECH (AI&DS)

STUDENT NM-ID: aut112823AIDS02

ROLL NO: 112823243001

DATE: 09/05/2025

Completed the project named as

QUALITY CONTOL IN MANUFACTURING

SUBMITTED BY,

NAME: Abisri U

MOBILE NO: 8122891716

Phase 4: Performance of the Project
Title: Smart Quality Control System for Manufacturing

Objective:

Phase 4 aims to enhance the performance of the Smart Quality Control System by refining defect detection algorithms, optimizing data processing pipelines, and ensuring the system can scale efficiently across larger manufacturing environments. This phase also focuses on improving sensor integration, reducing inspection time, and ensuring data security, while establishing the foundation for predictive quality control.

1. Defect Detection Performance Enhancement

Overview:

The machine vision and AI algorithms used for defect detection will be fine-tuned using real-world production data. The goal is to improve detection accuracy for both common and rare defects across various product types and materials.

Performance Improvements:

- **Algorithm Refinement:** Retraining the defect detection model using larger, more diverse datasets to improve the recognition of subtle or previously missed defects.
- **Hyper parameter Optimization:** Fine-tuning model parameters and applying pruning techniques to enhance both accuracy and processing speed.

Outcome:

The system will deliver more precise defect identification, significantly reducing false positives and false negatives, leading to improved product quality and lower rejection rates.

2. Inspection Speed and System Responsiveness

Overview:

This phase will improve the system's ability to perform high-speed inspections without sacrificing accuracy. Enhancements will be made to processing pipelines and hardware-software communication.

Key Enhancements:

- **Real-Time Processing:** Optimization of image and sensor data handling to ensure near-instantaneous feedback during production.
- **Parallel Processing:** Leveraging multi-threading and edge computing to process multiple inspection points simultaneously.

Outcome:

The system will support faster inspection cycles, enabling manufacturers to maintain production speed while ensuring consistent quality control.

3. Sensor Integration and IoT Connectivity

Overview:

Sensor networks, including visual cameras, temperature sensors, and vibration monitors, will be optimized for seamless real-time data flow and integration into the central control system.

Key Enhancements:

- **Improved Data Synchronization:** Calibrating sensors to synchronize data capture and processing across all checkpoints in the production line.
- **IoT Optimization:** Enhancing connectivity with existing factory infrastructure, including PLCs and SCADA systems.

Outcome:

Phase 4 will ensure seamless, real-time monitoring and feedback across multiple sensor types, increasing overall reliability of quality checks.

4. Data Security and System Reliability

Overview:

As the system scales across production lines, data security and system uptime become critical. Phase 4 will reinforce security protocols and improve fault tolerance. As industrial systems move toward automation and AI-driven decision-making, ensuring the confidentiality, integrity, and availability of data becomes paramount. In manufacturing environments, quality control data can include sensitive product specifications, defect patterns, operational timestamps, and employee access logs all of which must be protected from data leaks, tampering, or unauthorized access.

Key Enhancements:

- **Secure Data Transmission**: Implementing encrypted communication protocols (e.g., TLS/SSL) for data exchange between sensors, controllers, and the cloud.
- **Reliability Testing:** Conducting fault injection and recovery tests to evaluate system robustness under potential hardware or network failures.
- **Regular Security Audits:** Periodic vulnerability assessments and penetration testing will be conducted to identify and fix security gaps before they can be exploited.

Outcome:

The system will maintain operational continuity and data integrity even during high loads or partial system failures, meeting industrial cybersecurity and quality compliance standards.

5. System Performance Testing and Benchmarking

Overview:

A series of performance tests will validate the system's ability to operate under real-world production conditions. Benchmarking will cover processing speed, detection accuracy, and system stability.

Implementation:

- **Stress Testing:** Simulating high-throughput scenarios to measure the system's maximum processing capacity.
- **Benchmark Reports:** Collecting metrics like inspection time per unit, defect detection rate, and downtime statistics.
- **User Feedback Loop:** Gathering input from factory operators to improve the usability and reliability of the system interface.

Outcome:

The system will be validated for deployment in full-scale manufacturing environments, with performance metrics showing strong capability in both speed and precision.

Key Challenges in Phase 4

1. **Scalability in Production Environments**
    - **Challenge:** Maintaining performance as the system scales across more machines or production lines.
    - **Solution:** Distributed architecture and edge computing support to ensure load balancing and real-time performance.

2. **Sensor Accuracy and Calibration**
    - **Challenge:** Ensuring consistent and accurate sensor data across different machines and environmental conditions.
    - **Solution:** Automated sensor calibration routines and real-time error correction algorithms.

3. **System Downtime Risk:**
    - **Challenge:** Preventing system failures that could disrupt production.
    - **Solution:** Implementing redundancy and failover mechanisms along with continuous health monitoring.

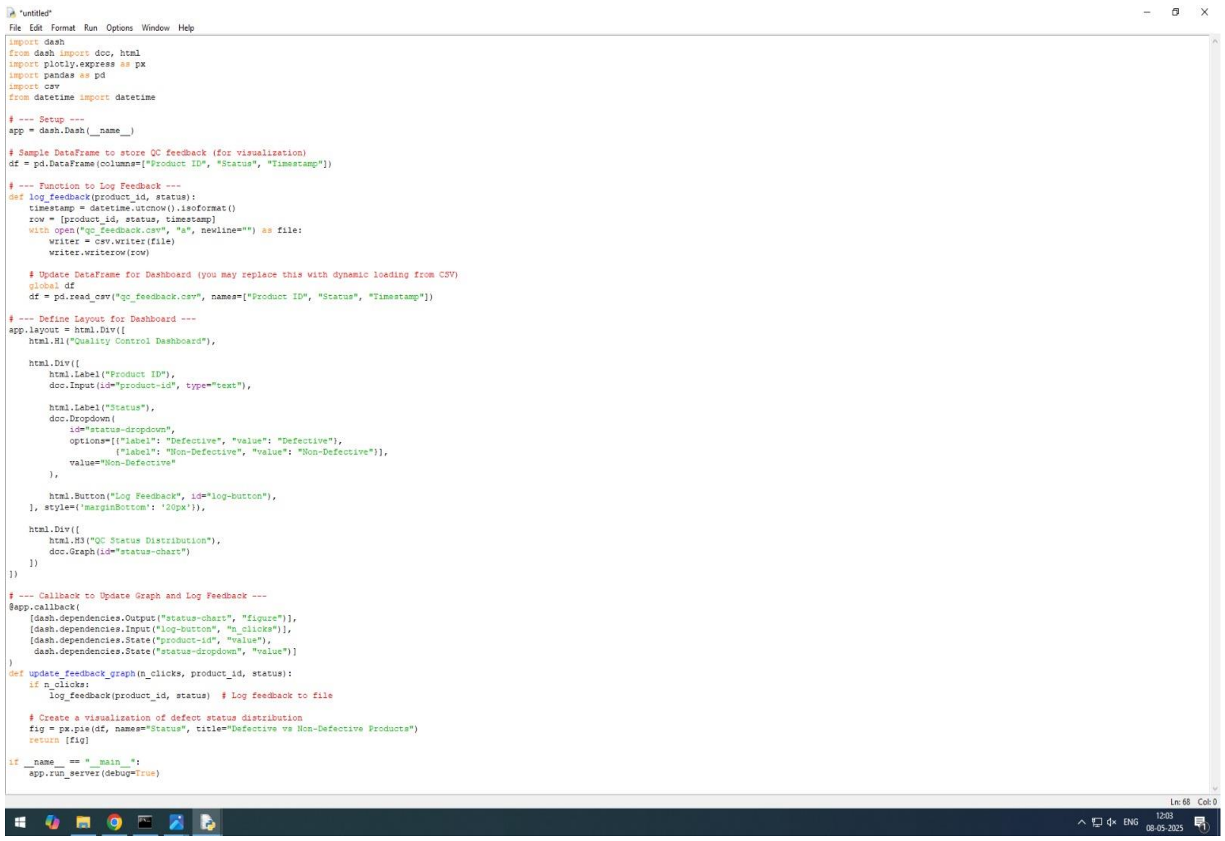
Outcomes of Phase 4

1. **Improved Defect Detection:** The AI model will identify a wider range of defects more accurately and consistently.
2. **Faster Inspection Speed:** Real-time processing enables seamless integration with high-speed production lines.
3. **Enhanced Sensor Integration:** Multi-sensor data will be synchronized and processed efficiently for better insights.
4. **Robust Security and Reliability:** Data will be securely stored and transmitted, with fault-tolerant architecture in place.

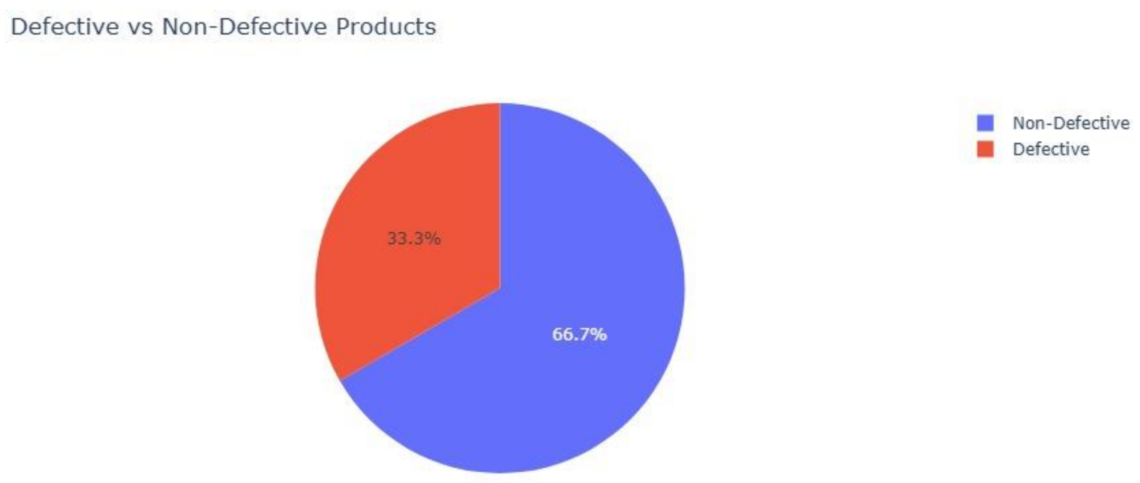
Next Steps for Finalization:

The final phase will involve full system deployment on the manufacturing floor, gathering live production feedback, and conducting final tuning for performance and accuracy. The project will also prepare documentation and training materials for end-user adoption.

Sample Code for Phase 4:



```python
import dash
from dash import dcc, html
import plotly.express as px
import pandas as pd
import csv
from datetime import datetime

# --- Setup ---
app = dash.Dash(__name__)

# Sample DataFrame to store QC feedback (for visualization)
df = pd.DataFrame(columns=["Product ID", "Status", "Timestamp"])

# --- Function to Log Feedback ---
def log_feedback(product_id, status):
    timestamp = datetime.utcnow().isoformat()
    row = [product_id, status, timestamp]
    with open("qc_feedback.csv", "a", newline="") as file:
        writer = csv.writer(file)
        writer.writerow(row)

    # Update DataFrame for Dashboard (you may replace this with dynamic loading from CSV)
    global df
    df = pd.read_csv("qc_feedback.csv", names=["Product ID", "Status", "Timestamp"])

# --- Define Layout for Dashboard ---
app.layout = html.Div([
    html.H1("Quality Control Dashboard"),

    html.Div([
        html.Label("Product ID"),
        dcc.Input(id="product-id", type="text"),

        html.Label("Status"),
        dcc.Dropdown(
            id="status-dropdown",
            options=[{"label": "Defective", "value": "Defective"},
                     {"label": "Non-Defective", "value": "Non-Defective"}],
            value="Non-Defective"
        ),

        html.Button("Log Feedback", id="log-button"),
    ], style={'marginBottom': '20px'}),

    html.Div([
        html.H3("QC Status Distribution"),
        dcc.Graph(id="status-chart")
    ])
])

# --- Callback to Update Graph and Log Feedback ---
@app.callback(
    [dash.dependencies.Output("status-chart", "figure")],
    [dash.dependencies.Input("log-button", "n_clicks")],
    [dash.dependencies.State("product-id", "value"),
     dash.dependencies.State("status-dropdown", "value")]
)
def update_feedback_graph(n_clicks, product_id, status):
    if n_clicks:
        log_feedback(product_id, status)  # Log feedback to file

    # Create a visualization of defect status distribution
    fig = px.pie(df, names="Status", title="Defective vs Non-Defective Products")
    return [fig]

if __name__ == "__main__":
    app.run_server(debug=True)
```


Defective vs Non-Defective Products
Non-Defective
Defective