

Phase 3: Implementation of Project

Title: Quality Control in Manufacturing

Objective:

The objective of Phase 3 is to implement the core components of the intelligent Quality Control System for manufacturing. This includes deploying AI algorithms for defect detection, integrating IoT sensors for real-time data collection, developing a user dashboard, and ensuring secure data handling. This phase transforms theoretical designs into functional components that enhance product quality, reduce waste, and optimize production efficiency.

1. AI-Powered Defect Detection

Overview:

The AI module will be responsible for detecting defects in manufactured goods using image recognition and sensor data. The system learns to identify common and uncommon defects using a dataset of labeled product images.

Implementation:

- **Image Classification Model:** A Convolutional Neural Network (CNN) is trained on labelled images of defected and non-defected products.
- **Sensor Fusion:** Real-time data from vibration, temperature, and pressure sensors will complement visual inspections to improve detection accuracy.
- **Threshold Logic:** The model raises alerts when defect probability exceeds pre-defined limits.

Outcome:

AI system will identify surface flaws, dimension errors, and assembly defects with high accuracy, reducing human error and speeding up inspection processes.

2. IoT Sensor Integration

Overview:

IoT devices are integrated to monitor production line parameters like temperature, humidity, and machine vibration, which influence product quality.

Implementation:

- **Sensor Network Setup:** Sensors are placed at critical points on the assembly line.
- **Data Collection:** Sensors transmit real-time data to a cloud-based platform for analysis.
- **Alerts & Actuation:** Automated alerts are generated when conditions go out of acceptable bounds.

Outcome:

The system supports preventive maintenance and quality assurance by continuously monitoring environmental and operational parameters.

3. Dashboard and Visualization

Overview:

A user-friendly dashboard is developed to provide real-time insights into product quality, production anomalies, and defect statistics.

Implementation:

- **Dashboard UI:** Built using React JS or Python Dash, showing live charts and system status.
- **Analytics Module:** Displays trends in defect rates, machine health, and operator performance.
- **User Controls:** Allows engineers to adjust AI sensitivity and set sensor thresholds.

Outcome:

The dashboard enhances transparency, enabling supervisors to make quick and informed decisions based on live production data.

4. Data Security Implementation

Overview:

Due to the sensitive nature of manufacturing data, this phase includes basic data security implementations. Additionally, strong data protection measures are necessary to comply with industry standards and regulations,

Implementation:

- **Encryption:** All collected data is encrypted using AES-256 standard.
- **Access Control:** Role-based access ensures that only authorized personnel can view or control system settings.
- **Secure Data Storage:** Data is stored on encrypted drives with regular backups to prevent loss in case of system failure
- **Data Masking:** Sensitive fields(e.g., operator IDs or IP addresses) are masked when shared for analysis or debugging purposes

Outcome:

Data remains secure and protected against unauthorized access, ensuring integrity and confidentiality.

5. Testing and Feedback Collection

Overview:

Initial testing of the system will be conducted to evaluate performance, user experience, and accuracy.

Implementation:

- **Test Runs:** Conducted on a small batch of products to identify false positives/negatives.
- **Feedback Loop:** Engineers and quality managers provide insights to improve model performance and usability.
- **Real-time Monitoring:** Sensors and dashboards tested continuously during production-like conditions for responsiveness and data accuracy.

Outcome:

Feedback guides optimization for the next phase, focusing on higher precision and better human-machine interaction.

Source Code:

```
#!/usr/bin/env python3
# Quality Control Dashboard (QC Dashboard)
# This script simulates a dashboard for monitoring product quality and collecting feedback.
# It uses a Flask web application and a SQLite database.

import os
import sys
import random
import datetime
import pandas as pd
import numpy as np
from flask import Flask, render_template, request, jsonify
from flask_sqlalchemy import SQLAlchemy
from flask_socketio import SocketIO, emit, join_room, leave_room
from flask_login import LoginManager, login_user, logout_user, login_required
from flask_wtf import FlaskForm
from wtforms import StringField, SubmitField
from wtforms.validators import Length, DataRequired

# Database Setup
db = SQLAlchemy(app)

# Models
class Product(db.Model):
    __tablename__ = 'products'
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    status = db.Column(db.String(20), nullable=False)
    timestamp = db.Column(db.DateTime, nullable=False)

class Feedback(db.Model):
    __tablename__ = 'feedback'
    id = db.Column(db.Integer, primary_key=True)
    product_id = db.Column(db.Integer, nullable=False)
    status = db.Column(db.String(20), nullable=False)
    timestamp = db.Column(db.DateTime, nullable=False)

# Flask App Setup
app = Flask(__name__)
app.config['SECRET_KEY'] = 'your_secret_key'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///qc_dashboard.db'
db.init_app(app)

# Routes
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/products')
def products():
    products = Product.query.all()
    return render_template('products.html', products=products)

@app.route('/feedback')
def feedback():
    feedbacks = Feedback.query.all()
    return render_template('feedback.html', feedbacks=feedbacks)

@app.route('/api/products', methods=['GET'])
def get_products():
    products = Product.query.all()
    return jsonify([product.to_dict() for product in products])

@app.route('/api/feedback', methods=['GET'])
def get_feedback():
    feedbacks = Feedback.query.all()
    return jsonify([feedback.to_dict() for feedback in feedbacks])

@app.route('/api/feedback', methods=['POST'])
def add_feedback():
    data = request.json
    product_id = data['product_id']
    status = data['status']
    timestamp = datetime.datetime.utcnow()
    feedback = Feedback(product_id=product_id, status=status, timestamp=timestamp)
    db.session.add(feedback)
    db.session.commit()
    return jsonify({'message': 'Feedback added successfully'})

# SocketIO Setup
socketio = SocketIO(app)

# Real-time Monitoring
def monitor_products():
    while True:
        products = Product.query.all()
        for product in products:
            # Simulate status change
            status = random.choice(['Defective', 'Non-Defective'])
            product.status = status
            db.session.commit()
            # Emit status change to clients
            socketio.emit('product_status_change', {'product_id': product.id, 'status': status})
        # Simulate new feedback
        product_id = random.choice([product.id for product in products])
        status = random.choice(['Defective', 'Non-Defective'])
        timestamp = datetime.datetime.utcnow()
        feedback = Feedback(product_id=product_id, status=status, timestamp=timestamp)
        db.session.add(feedback)
        db.session.commit()
        # Emit new feedback to clients
        socketio.emit('new_feedback', {'feedback_id': feedback.id, 'product_id': product_id, 'status': status, 'timestamp': timestamp})
        # Sleep for 10 seconds
        time.sleep(10)

# Start Monitoring
monitor_products()

# Run the app
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

Challenges and Solutions:

1. Model Accuracy

- **Challenge:** Initial misclassifications due to limited dataset.
- **Solution:** Implement continuous learning and regular dataset augmentation.

2. Sensor Calibration

- **Challenge:** Sensor drift and inaccurate readings.
- **Solution:** Regular calibration schedules and redundancy in sensor placement.

3. User Adoption

- **Challenge:** Resistance from traditional quality control teams.
- **Solution:** Provide training sessions and demonstrate productivity benefits.

Outcomes of Phase 3:

By the end of Phase 3, the following milestones will be achieved:

1. A functional AI-based defect detection system.
2. Real-time IoT integration for environmental and machine monitoring.
3. A secure, interactive dashboard for quality control insights.
4. Successful test runs with high defect detection accuracy (>90%).