Experiment 9

Student Name: Sanjusha Singh **UID: 22BCS13130**
Branch: CSE **Section/Group: 614-B**
Semester: 6th **Date of Performance: 17/4/25**
Subject Name: Cloud IoT **Subject Code: 22CSP-367**

1. **Aim:** Automate quality inspection of products using cameras and edge computing.

2. **Objective:** To design and implement an automated quality inspection system for products using cameras and edge computing.

3. **Hardware / Software Used:** IoT Cameras, Edge Computing Devices, Cloud Integration (Optional), Actuators and Alerts.

4. **Procedure:**

a. Data Collection (Image Acquisition)

- Cameras: Use industrial/machine vision cameras placed along the production line.
- Lighting: Optimize lighting to reduce shadows and enhance defect visibility.
- Trigger Mechanism: Use sensors to capture images as products pass by. b. Image Preprocessing
- Enhancement: Remove noise and highlight key features.
- Resizing: Standardize image size (e.g., 224x224 pixels).
- Normalization: Scale pixel values.
- Data Augmentation: Apply rotations, lighting variations, etc., for training diversity.

c. Defect Detection Model

- CNNs: Use CNNs for image-based defect classification.
- Pre-trained Models: Fine-tune models like ResNet or VGG16 on labeled datasets.
- Edge Inference: Deploy the model on edge devices for real-time detection.

d. Edge Computing

- Real-Time Processing: Analyze images locally to reduce latency.
- Hardware: Use Raspberry Pi, NVIDIA Jetson, or Intel NUC based on performance needs.
- Model Optimization:
- Quantization: Reduce weight precision (e.g., float32 → int8).
- Pruning: Remove unnecessary model components.

e. Integration with Actuators

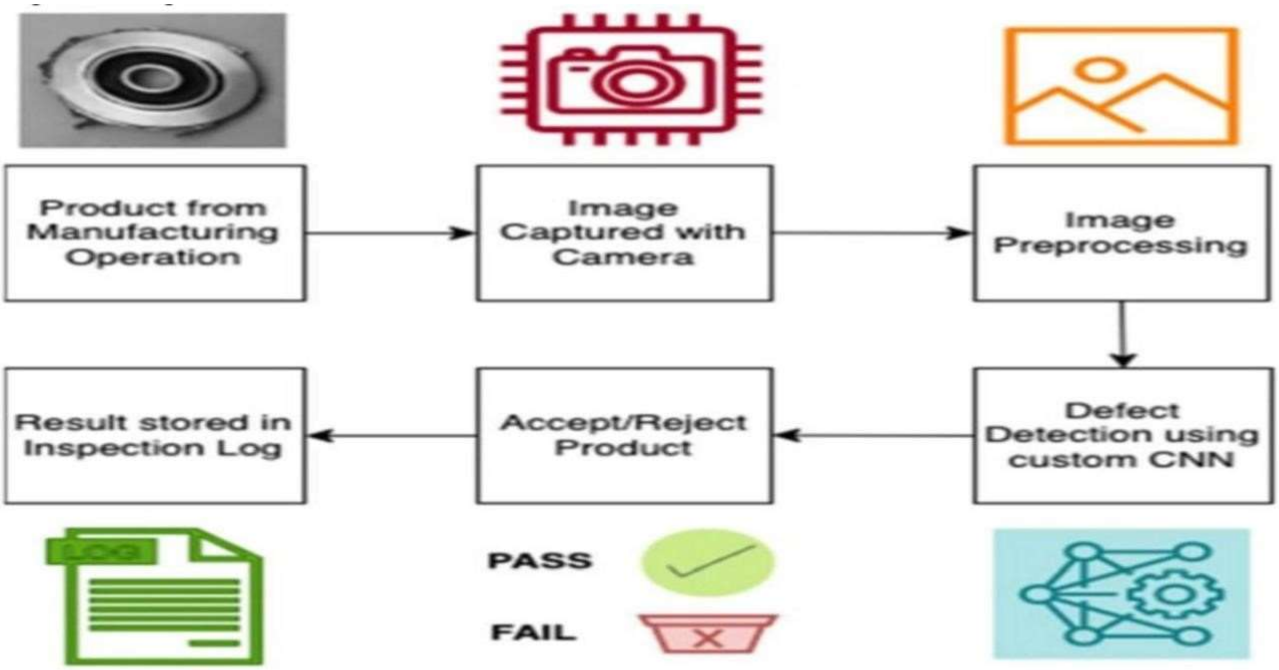
- Defect Rejection: Trigger actuators to remove defective products. • Alerts: Notify operators via SMS, email, or IoT dashboards.

5. Code:

```
import cv2 import
numpy as np import
tensorflow as tf
from tensorflow.keras.applications import VGG16
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten

# Load and preprocess image image =
cv2.imread("product.jpg") image_resized = cv2.resize(image,
 (224, 224)) image_normalized = image_resized / 255.0

# Expand dimensions to match model input shape input_image
= np.expand_dims(image_normalized, axis=0)

# Load trained model (already trained and saved as .h5)
model = tf.keras.models.load_model("defect_detection_model.h5")

# Make prediction prediction =
model.predict(input_image) class_index
 = np.argmax(prediction)

# Interpret and act on prediction
if class_index == 0:
print("Product is defective")
   send_alert("Defective product detected!")
   trigger_actuator()
else:
       print("Product is not defective")
```

6.

Result:



7. Learning Outcomes:

1. Understand how to preprocess images (resizing, normalization) and feed them into a deep learning model for real-time classification.

2. Gain practical experience in using transfer learning by fine-tuning a pre-trained VGG16 model for a specific task like defect detection.

3. Learn how to integrate machine learning models with alert systems and actuators to enable automated decision-making in industrial environments.