

REVOLUTIONIZING TOMATO PRODUCTION AND QUALITY ASSESSMENT USING AI

Herath H.M.R.K

IT20665548

B.Sc. (Hons) in Information Technology Specialized in Information
Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

January 2024

REVOLUTIONIZING TOMATO PRODUCTION AND QUALITY ASSESSMENT USING AI

Herath H.M.R.K

IT20665548

B.Sc. (Hons) in Information Technology Specialized in Information
Technology


Department of Information Technology

Sri Lanka Institute of Information Technology

January 2024

DECLARATION OF THE CANDIDATE & SUPERVISOR

I declare that this is my own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Herath H.M.R.K	IT20665548	

The above candidate is carrying out research for the undergraduate Dissertation under my supervision

Signature of the supervisor

Date

.....

09.04.2025

(Mr. Amila Senarathna)

ABSTRACT

Ensuring the quality and market readiness of tomatoes is a critical challenge in the agricultural sector. Current grading systems primarily rely on manual inspection, which is inherently subjective, inconsistent, and labor-intensive, making it inefficient for large-scale operations. These methods often fail to identify subtle defects or account for environmental factors such as lighting variations and background noise. As the demand for high-quality produce increases, there is a pressing need for automated systems that can accurately and consistently assess tomato quality. This research aims to address these challenges by developing an integrated system for Tomato Defect Detection and Quality Grading using advanced machine learning models.

consists of two key components:

1. Defect Detection: Leveraging EfficientNet to identify defects such as cracks, shriveling, and rot with high precision. EfficientNet's scalability and accuracy make it ideal for detecting both prominent and subtle defects under diverse conditions.
2. Quality Grading: Utilizing MobileNetV2, a lightweight yet powerful model, to classify tomatoes into predefined commercial grades (Grade A, B, C, and Rejected). MobileNetV2's efficiency ensures rapid grading without compromising accuracy, making it suitable for real-time applications.

The system is trained on a comprehensive dataset containing annotated images of tomatoes with varying defect types and quality attributes. Data preprocessing techniques such as normalization and augmentation are applied to enhance robustness and address class imbalance. Both models are fine-tuned to optimize performance, achieving high accuracy and consistency across training and validation datasets. This solution represents a significant advancement in agricultural AI by providing a scalable and objective framework for quality assessment. The integration of EfficientNet and MobileNetV2 ensures accurate defect detection and precise grading, even in challenging environments. Additionally, the system is designed to support potential deployment in real-world scenarios, including IoT-based field operations and mobile applications for farmers and supply chain managers.

By automating the defect detection and quality grading processes, this research not only reduces the reliance on manual labor but also minimizes errors, enhances productivity, and ensures uniformity in quality standards. Furthermore, the proposed system aligns with the broader goal of leveraging technology to promote sustainable and efficient agricultural practices.

Keywords: Tomato Quality Assessment, Defect Detection, EfficientNet, MobileNetV2, Machine Learning, Agricultural AI, Image Classification, Quality Grading, Automation in Agriculture.

ACKNOWLEDGEMENT

The Sri Lanka Institute of Information Technology (SLIIT) gave me the chance and assistance to complete my senior research project, "Revolutionizing Tomato Production and Quality Assessment Using AI," for which I am truly grateful. A major component of both my intellectual and personal development has been this effort. My supervisor, Mr. Amila Senarathna, has provided invaluable direction, constructive criticism, and continuous support throughout this research, for which I am incredibly grateful. The success of this initiative and its direction were greatly influenced by his mentoring. Also deserving of my sincere gratitude is my co-supervisor, Dr. Lakmini Abeywardhana, whose technical know-how, wise counsel, and unwavering support improved the caliber and applicability of my work. My appreciation also goes out to the faculty, technical staff, and administrative team at SLIIT for providing the tools, facilities, and academic atmosphere needed to successfully finish this project.

We are especially grateful to the farmers, data contributors, and agricultural specialists whose perspectives and hands-on experience in tomato cultivation helped us comprehend the real-world difficulties and modify this AI-based solution appropriately. I want to express my sincere gratitude to my family and friends for their tolerance, inspiration, and moral support throughout trying times. Their faith in me served as a continual source of encouragement. Last but not least, I want to thank everyone who helped this project be completed successfully, whether directly or even indirectly. I have improved my technical and problem-solving abilities as a result of this research, and it has strengthened my resolve to use contemporary technology to create significant solutions for sustainable agriculture.

Table of Contents

Contents

DECLARATION OF THE CANDIDATE & SUPERVISOR	3
ABSTRACT	4
1. INTRODUCTION	9
1.1. Background & Literature survey.....	9
1.2. Research Gap	10
1.3. Research Problem.....	12
1.4. Proposed Project	13
1.5. Data Collection and Preprocessing	13
1.6. Research Objectives	14
1.6.1. Main Objective	14
1.6.2. Specific Objectives.....	14
2. METHODOLOGY	16
2.1. Methodology	16
2.1.1. Conceptual Framework	17
2.1.2. Data Collection.....	18
2.1.3. Data Preprocessing	21
2.1.4. Model Development	24
2.1.5. Model Training and Validation	25
2.1.6. System Integration.....	27
2.1.7. Interface and Usability Design	28
2.1.8. Deployment	30
2.2. Commercialization Aspects of the Product.....	32

2.2.1. Target Market	32
2.2.2. Business Model	33
2.2.3. Monetization Plan.....	34
2.2.4. Competitive Advantage	34
2.2.5. Future Scope.....	35
2.3. Testing and Implementation.....	36
2.3.1. Testing Phases	36
2.3.2. Evaluation Metrics	36
2.3.3. Real-World Testing	37
2.3.4. Feedback & Iteration	38
2.3.5. Final Deployment	39
3. RESULTS AND DISCUSSION	40
3.1. Results 40	
3.1.1. Defect Detection (EfficientNet)	40
3.1.2. Quality Grading (MobileNetV2)	41
3.1.3. Latency Performance.....	42
3.2. Research Findings	42
3.3. Discussion	43
3.3.1. Limitations.....	43
3.3.2. Future Opportunities.....	43
4. Conclusions	44
5. Reference.....	46
6. Glossary	48
7. APPENDICES	49

LIST OF FIGURES

Figure 1 NOVELTY OF THE PROPOSED SYSTEM	11
Figure 2 System overview	16
Figure 3 Data Collection.....	19
Figure 4 Data Collection.....	19
Figure 5 Data Collection.....	20
Figure 6 Data Preprocessing	21
Figure 7 Data Preprocessing	22
Figure 8 Data Preprocessing	23
Figure 9 Model Development	25
Figure 10 Model Training and Validation.....	26
Figure 11 Model Training and Validation.....	27
Figure 12 System Integration.....	28
Figure 13 Interface and Usability Design	29
Figure 14 Interface and Usability Design	30
Figure 15 Interface and Usability Design	31
Figure 16 DEPLOYMENT.....	32
Figure 17 UI SCREENSHOTS.....	50
Figure 18 UI SCREENSHOTS.....	50
Figure 19 Gantt chart	51
Figure 20 Work Breakdown Structure	52
Figure 21 Plagiarism Report.....	52

LIST OF TABLES

Table 1 Tomato Defect Categories.....	18
Table 2 Dataset Summary	20
Table 3 EfficientNet Model Performance Comparison	40
Table 4 MobileNetV2 Model Performance Comparison.....	41

1. INTRODUCTION

1.1. Background & Literature survey

In the agricultural sector, quality assessment and grading of produce play a crucial role in ensuring market standards and customer satisfaction. Tomatoes, as one of the most widely consumed agricultural products, require meticulous grading processes to meet consumer and commercial expectations. Traditionally, these processes have relied on manual inspection, which is inherently subjective, labor-intensive, and inconsistent, especially for large-scale operations. Research has shown that such methods often fail to detect subtle defects and lack the scalability to cater to modern agricultural demands. Advancements in machine learning and computer vision have paved the way for automated systems that address these limitations. Studies by researchers such as Zhang et al. [1] demonstrate the efficacy of EfficientNet in detecting visual anomalies in agricultural products. Automated defect detection systems leverage these technologies to identify defects like cracks, shriveling, and rot with greater precision and reliability compared to manual methods. Quality grading, another critical component, requires systems that classify produce based on predefined standards, such as ripeness, size, and texture. MobileNetV2, as highlighted by Sandler et al. [2], offers a lightweight yet powerful framework suitable for classification tasks, making it an ideal choice for real-time grading applications. The integration of such models ensures scalability and consistency in grading outcomes. However, despite these advancements, challenges persist. Variability in environmental conditions, such as lighting and background noise, often affects the accuracy of defect detection and grading models. Additionally, the lack of standardized and diverse datasets for training machine learning models further limits their applicability. Recent efforts, such as those by Patel et al. [3], emphasize the importance of robust preprocessing techniques and data augmentation to mitigate these issues. This research builds on existing methodologies by integrating EfficientNet for defect detection and MobileNetV2 for quality grading. The proposed system aims to enhance accuracy and scalability, addressing real-world challenges while aligning with commercial standards. By automating these processes, the system seeks to revolutionize tomato quality assessment, benefiting both producers and consumers.

1.2. Research Gap

While advancements in agricultural automation have made significant strides in addressing traditional challenges, a notable gap persists in creating comprehensive systems for tomato defect detection and quality grading that are both scalable and adaptable to real-world conditions. Existing approaches largely focus on individual aspects, such as defect detection or quality grading, but rarely integrate both into a unified framework that aligns with commercial standards.

1.2.1. Key Gaps Identified

Manual Limitations: Traditional grading relies on manual inspection, which is prone to human error, inconsistency, and scalability challenges in large-scale operations. Subtle defects such as small cracks or early signs of rot are often missed, leading to inefficiencies in quality assessment.

Lack of Integration: Current systems often handle either defect detection or grading independently, failing to provide an end-to-end solution for tomato quality assessment. There is limited research on combining defect detection with commercial-grade classification in a cohesive system.

Dataset Challenges: Publicly available datasets for defect detection and grading are scarce, imbalanced, or lack variability, which hampers model training and real-world applicability. Existing datasets often fail to account for environmental variations such as lighting conditions, occlusions, and complex backgrounds.

Technological Constraints: Many models are computationally intensive, making them unsuitable for deployment in real- time or resource-constrained environments like farms or mobile devices. Current systems struggle with generalizing across diverse tomato varieties and environmental conditions.

Proposed Solution: The proposed system bridges these gaps by: Utilizing EfficientNet for robust defect detection, capable of identifying subtle defects under diverse conditions. Employing MobileNetV2 for lightweight, accurate quality grading aligned with commercial standards. Integrating both components into a unified framework to deliver consistent, scalable, and efficient quality assessment. Developing a comprehensive dataset with diverse defect types and quality grades, incorporating augmentation techniques to enhance robustness. Ensuring real-time operability by optimizing models for deployment on IoT platforms and mobile devices.

Features	Proposed System	Existing Systems
Tomato Defect Detection	✓	✓
Quality Grading	✓	✗
Texture Analysis	✓	✗
Commercial Standard Classification	✓	✗
AI Integration for Real-Time Detection	✓	✗
IoT Connectivity	✓	✗
Cloud-based Data Storage	✓	✓
User-friendly Dashboard	✓	✗

FIGURE 1 NOVELTY OF THE PROPOSED SYSTEM

The research gap lies in the absence of a unified framework that integrates tomato defect detection and quality grading into a single, efficient, and scalable system. Existing methods either focus on defect detection or grading in isolation, lacking the ability to provide a comprehensive solution. Additionally, current systems struggle with real-world variability, such as environmental conditions and diverse tomato varieties, limiting their effectiveness in large-scale agricultural operations. The proposed system seeks to address this gap by developing a robust AI-driven framework that combines EfficientNet for defect detection and MobileNetV2 for quality grading. This integrated approach ensures precise identification of defects and accurate classification into commercial quality grades, offering a scalable, real-time solution that aligns with market standards and operational needs.

1.3. Research Problem

The The agriculture industry faces persistent challenges in ensuring the consistent quality and grading of tomatoes, a crop that significantly contributes to global food supply chains. Despite advancements in technology, the current methods for defect detection and quality grading remain inadequate, particularly when scaled to meet the demands of commercial production.

Limitations of Existing Methods Manual Inspection: Tomato quality assessment primarily relies on human inspectors, which is labor-intensive, prone to fatigue, and subject to inconsistencies. Subtle defects such as fine cracks, early rot, or uneven ripening are often overlooked, resulting in errors and reduced efficiency. Manual grading struggles to meet the demands of large-scale operations, particularly in fast-paced supply chains.

Automated Systems: Existing automated systems often focus on singular tasks, such as defect detection or quality grading, without providing an integrated solution. Many models lack the robustness to handle real-world variations, such as changes in lighting, background interference, and differences in tomato varieties. Limited availability of diverse and annotated datasets hampers the training of models, reducing their generalizability and accuracy.

Challenges in Technology Adoption

Computational Constraints: Advanced machine learning models often require significant computational power, making them less feasible for real-time or on-site deployment.

Scalability: Current solutions do not sufficiently address the need for scalability, particularly for small-scale farmers or decentralized operations.

Alignment with Standards: Grading systems often fail to align with commercial quality standards, limiting their adoption by supply chain stakeholders.

The Need for a Comprehensive Solution Given the importance of consistent quality in tomatoes for both producers and consumers, there is a pressing need for a reliable, scalable, and accurate system. This system should:

Detect defects such as cracks, shriveling, and rot with high precision using advanced machine learning techniques. Grade tomatoes into commercially recognized categories such as Grade A, B, C, and Rejected. Operate effectively under diverse real-world conditions, ensuring robust performance in different environmental scenarios.

1.4. Proposed Project

This project aims to develop a comprehensive AI-based system for Tomato Defect Detection and Quality Grading, addressing critical challenges in the agricultural sector. The proposed system will integrate cutting-edge machine learning technologies to enhance efficiency, accuracy, and scalability in tomato quality assessment. The key components include:

- Defect Detection Using EfficientNet** Develop a defect detection module using EfficientNet model optimized for image classification tasks. The system will identify defects such as cracks, shriveling, rot, and other visual anomalies with high precision and robustness. Real-time predictions will ensure faster and more reliable defect identification in large-scale operations.
- Quality Grading Using MobileNetV2** Implement a lightweight yet powerful grading module using MobileNetV2 to classify tomatoes into predefined grades: Grade A, Grade B, Grade C, and Rejected. Ensure alignment with commercial quality standards for consistent and objective grading. The model will assess attributes like size, texture, and ripeness while maintaining scalability for real-world applications

1.5. Data Collection and Preprocessing

Curate a comprehensive dataset comprising diverse tomato images representing various defect types and quality grades. Apply preprocessing techniques such as data augmentation, normalization, and resizing to improve model performance under different environmental conditions.

System Integration Combine the defect detection and quality grading modules into a unified pipeline to provide seamless quality assessment from raw image input to classification output. The integrated system will handle input variability, ensuring robust performance in different lighting conditions and environments.

Real-Time Deployment and Accessibility Design the system for potential integration with IoT-enabled devices and mobile applications to facilitate real-time deployment. Develop a user-friendly interface for farmers, supply chain stakeholders, and quality control personnel to access grading results efficiently.

1.6. Research Objectives

1.6.1. Main Objective

To develop an AI-based system for Tomato Defect Detection and Quality Grading that leverages advanced machine learning models, such as EfficientNet and MobileNetV2, to automate the classification of tomato defects and grades. The system aims to enhance accuracy, consistency, and scalability in quality assessment, addressing the limitations of traditional manual methods and supporting agricultural supply chain stakeholders with a robust, real-time solution.

1.6.2. Specific Objectives

Develop a Defect Detection Module: Implement an EfficientNet-based model to identify common tomato defects such as cracks, rot, and shriveling. Ensure high precision in defect classification under diverse environmental conditions, including varying lighting and backgrounds. **Design a Quality Grading System:** Utilize MobileNetV2 to classify tomatoes into predefined commercial grades (Grade A, Grade B, Grade C, and Rejected). Align the grading system with standardized quality benchmarks to ensure consistent outputs. **Curate and Preprocess Datasets:** Collect and annotate a comprehensive dataset representing various defects and quality grades. Apply preprocessing techniques like augmentation and normalization to address class imbalance and enhance model robustness.

Integrate Defect Detection and Grading Modules: Combine the two models into a unified framework capable of delivering both defect and grade classification in a single workflow. Ensure seamless data flow and processing from raw image input to final quality assessment output.

Optimize for Real-Time Deployment: Design the system for integration with IoT-enabled devices or mobile platforms to facilitate on-field usage by farmers and supply chain operators. Optimize computational efficiency to ensure fast processing and scalability.

Evaluate Model Performance: Conduct comprehensive testing using validation datasets and real-world tomato samples. Measure performance through metrics such as accuracy, precision, recall, and F1 score to validate system reliability.

Promote Usability and Accessibility: Develop a user-friendly interface for stakeholders to access defect and grading results efficiently. Provide deployment options suitable for small and large-scale agricultural operations, ensuring inclusivity.

Contribute to Sustainable Agriculture: Enhance productivity and reduce waste by ensuring consistent quality assessment of tomatoes. Support data-driven decision-making for farmers and supply chain stakeholders.

2. METHODOLOGY

2.1. Methodology

A comprehensive AI-based system development process, comprising image dataset collecting, preprocessing, model training with EfficientNet and MobileNetV2, and real-time deployment through an intuitive interface, is the methodology used in this study. Detecting flaws and grading tomato quality with deep learning technologies in an efficient, scalable, and field-ready solution is the main goal.

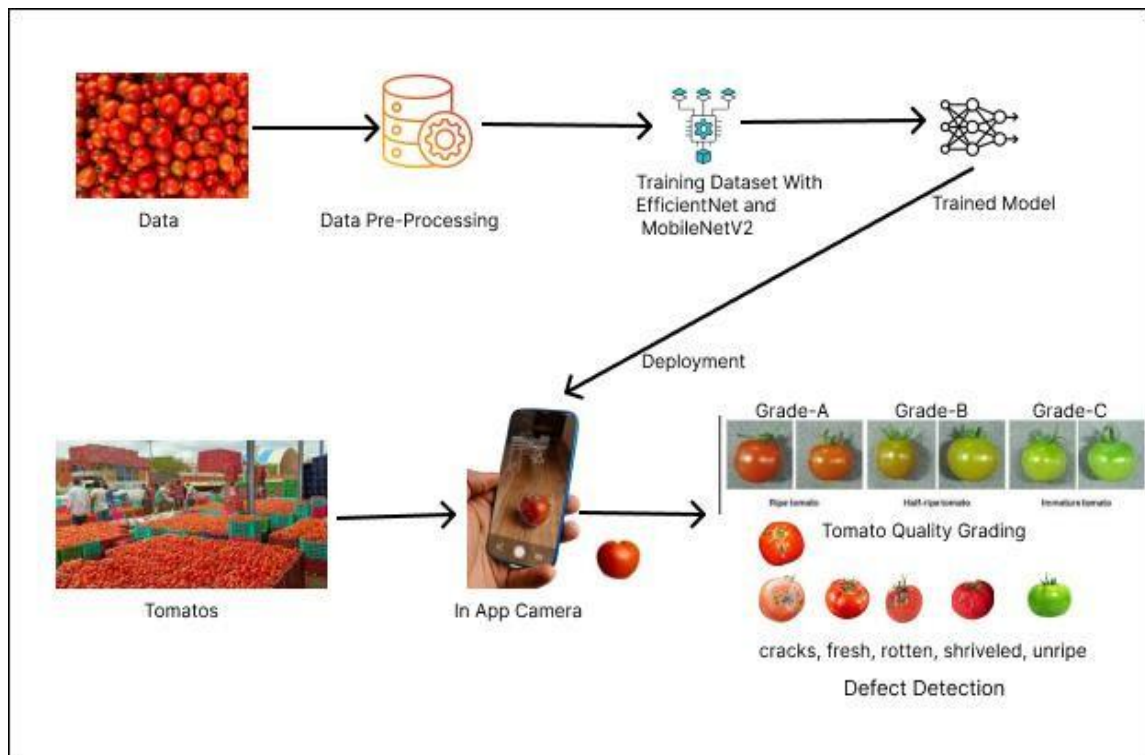


FIGURE 2 SYSTEM OVERVIEW

2.1.1. Conceptual Framework

The proposed system's conceptual framework describes a structured pipeline that analyzes tomato input photos to determine the commercial grade and physical state of the tomatoes. Two deep learning models' combined strengths are utilized in the design: EfficientNet: Used for highly accurate visual fault identification, including rot, fractures, and shriveling.

Tomatoes are categorized using MobileNetV2 according to surface characteristics and visual cues into predetermined quality classes (classes A, B, C, and Rejected).

Tomato samples are evaluated thoroughly and robustly thanks to this dual-model architecture. The conceptual framework's step-by-step procedure comprises:

- Image Input: A user interface or a camera-equipped device are used to take and upload tomato photos.
- Preprocessing aims to standardize format and increase model resilience by resizing, normalizing, and enhancing input photos using augmentation techniques.
- In order to identify any obvious flaws, the preprocessed image is first run through the EfficientNet model.
- Quality Grade Prediction: The MobileNetV2 model processes the image to compute the quality grade if defects are detected or if no serious defects are discovered.
- Result Output in Real Time: A responsive web or mobile interface shows the user the results from both models in real time.

2.1.2. Data Collection

Tomato photos were gathered from a variety of sources in order to provide a strong and varied dataset that could be used to train and validate the AI models:local markets and farms: Real-world photos of tomatoes with inherent flaws and variable quality levels were taken in various settings.Agricultural data archives: The photos came from reputable research centers and agricultural divisions that keep plant quality and disease classification records.Open-source datasets on the web: The data was further enriched using publicly accessible datasets from academic repositories and websites such as GitHub and Kaggle.

TABLE 1 TOMATO DEFECT CATEGORIES

Defect Type	Description	Example Symptoms
Cracks	Surface cracking due to irregular growth	Open splits on skin
Rot	Fungal or bacterial decay	Dark, mushy spots
Shriveled	Dehydration or aging	Wrinkled, sunken surface
Unripe	Not fully matured	Green patches, firm texture
Fresh	Ideal condition	Smooth, firm, red skin

Researchers and agricultural specialists individually labelled each photograph after a thorough assessment. Labeling was done according to two main standards:Sorts of Defects: visual flaws like Cracks: Surface divisions brought on by excessive ripening or erratic watering. Rot: Dark patches or discolorations brought on by bacterial or fungal infestations. Wrinkled skin that suggests dehydration or over-maturity is called "shriveling." Good quality Ratings: Grade A: Consistent in size, shape, and color; no obvious flaws. Grade B: Tiny color or form changes; one or a few minor flaws. Grade C: Processing-acceptable; noticeable distortions or mild flaws. Rejected: Extremely rotten or flawed; unfit for sale or consumption. This methodical approach to data gathering and annotation made sure that the training models were given high-quality, precisely labeled inputs that were representative of actual situations.

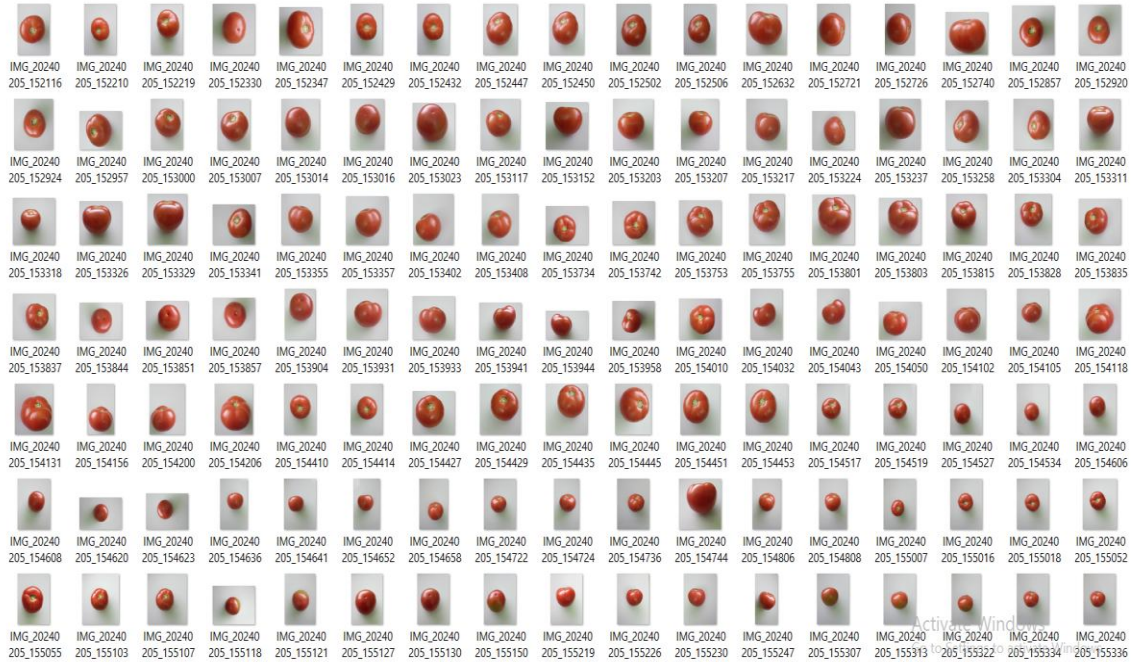


FIGURE 3 DATA COLLECTION

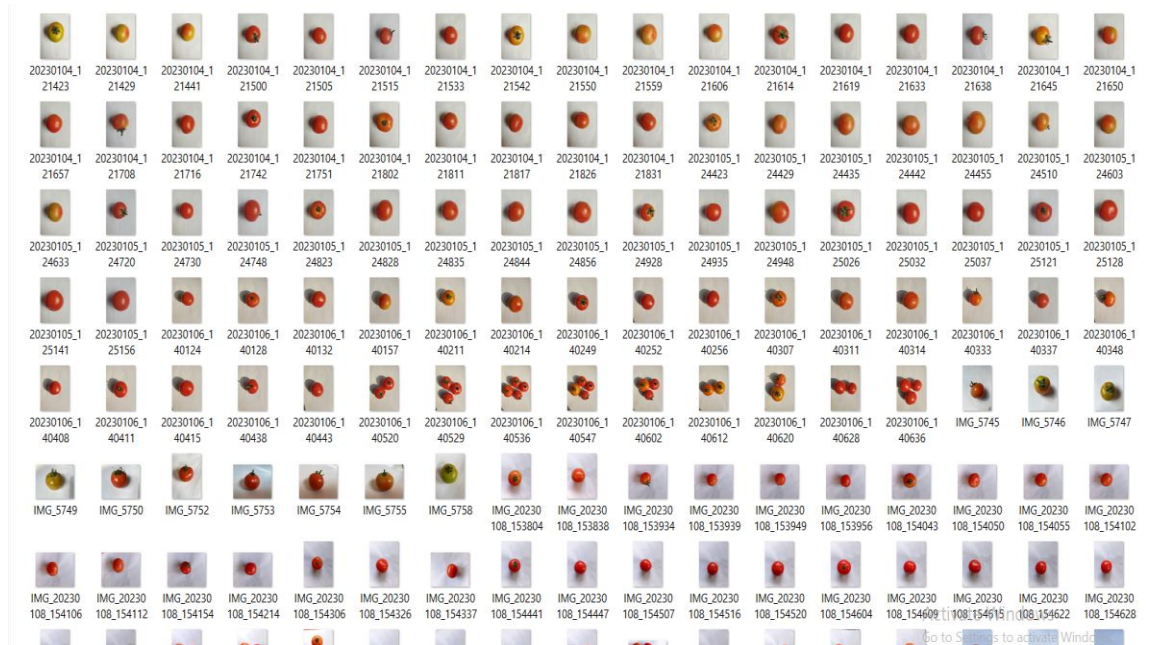


FIGURE 4 DATA COLLECTION



FIGURE 5 DATA COLLECTION

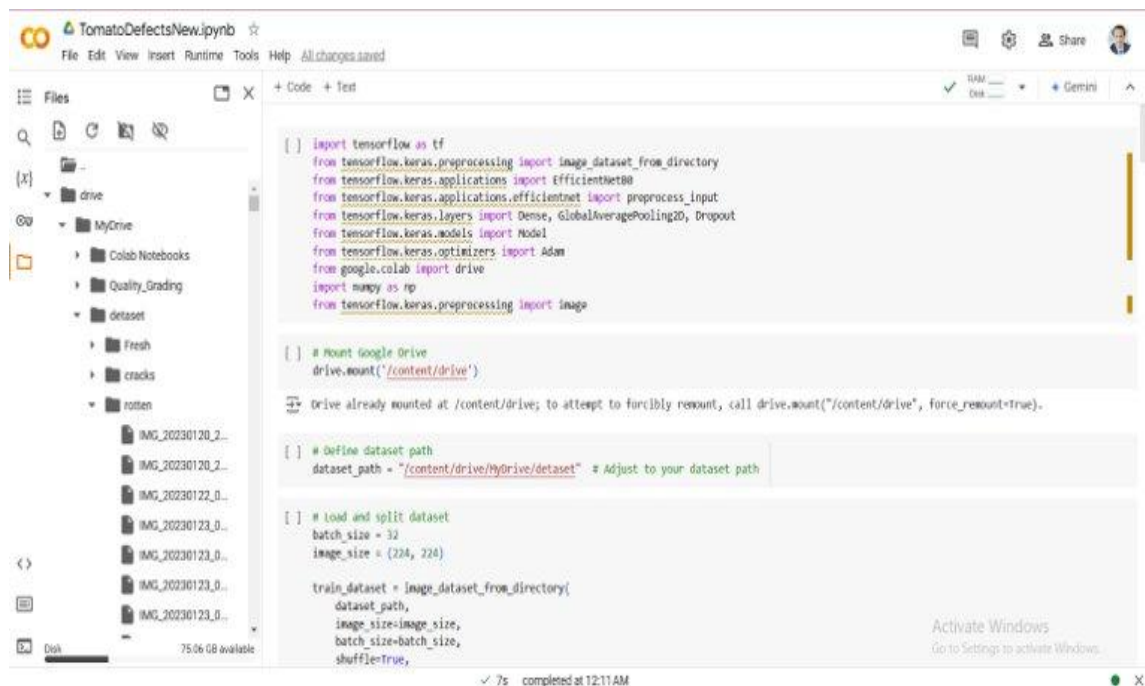
TABLE 2 DATASET SUMMARY

Dataset Type	Number of Images	Image Resolution	Annotation Tool Used
Training	1,200	224x224	Roboflow
Validation	300	224x224	Roboflow
Testing	200	224x224	Roboflow

2.1.3. Data Preprocessing

The dataset had to be properly preprocessed in order to maximize model performance during training. Steps were taken as follows: Resizing: In order to guarantee compatibility with the input dimensions needed by the EfficientNet and MobileNetV2 architectures, all tomato photos were uniformly scaled to 224x224 pixels. Standardization: To make the input distribution uniform, pixel values were scaled to the [0,1] range. Training became stable as a result, and the learning algorithm's convergence improved. In order to reduce overfitting and enhance dataset variety, a number of augmentation strategies were used: Rotate by ± 30 degrees to replicate various camera perspectives. Both vertical and horizontal flipping Variations in contrast and brightness arbitrary cropping and zooming

Balancing: enhanced duplicates were used to oversample minority classes due to the dataset's class imbalance (fewer highly faulty tomatoes, for example). By doing this, it was made sure that during training, the model would not start to favor majority classes. Format conversion: To speed up model input parsing, images were transformed into a standard format (such as JPEG or PNG) and tagged with a common naming scheme. The dataset was made much more robust by this preprocessing workflow, which also prepared it for high-performance training in a variety of lighting and ambient circumstances.



```
[ ] import tensorflow as tf
from tensorflow.keras.preprocessing import image_dataset_from_directory
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.applications.efficientnet import preprocess_input
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from google.colab import drive
import numpy as np
from tensorflow.keras.preprocessing import image

[ ] # Mount google Drive
drive.mount('/content/drive')

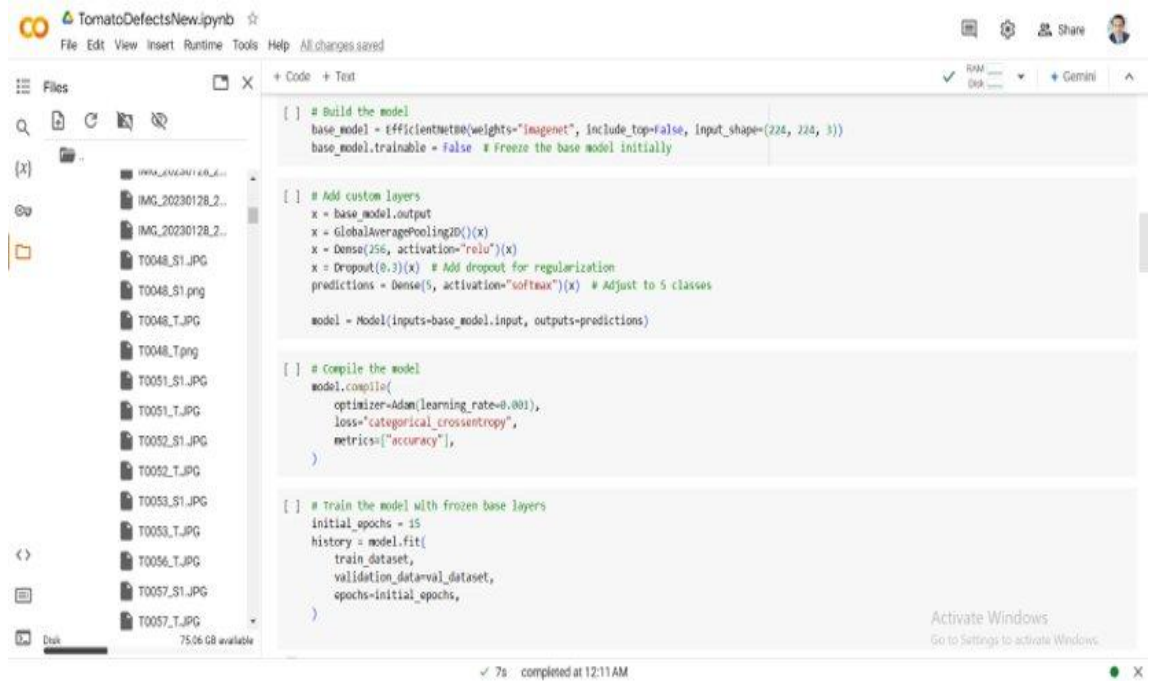
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[ ] # Define dataset path
dataset_path = "/content/drive/MyDrive/dataset" # Adjust to your dataset path

[ ] # Load and split dataset
batch_size = 32
image_size = (224, 224)

train_dataset = image_dataset_from_directory(
    dataset_path,
    image_size=image_size,
    batch_size=batch_size,
    shuffle=True,
```

FIGURE 6 DATA PREPROCESSING



```
[ ] # Build the model
base_model = EfficientNetB0(weights="imagenet", include_top=False, input_shape=(224, 224, 3))
base_model.trainable = False # Freeze the base model initially

[ ] # Add custom layers
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(256, activation="relu")(x)
x = Dropout(0.5)(x) # Add dropout for regularization
predictions = Dense(5, activation="softmax")(x) # Adjust to 5 classes

model = Model(inputs=base_model.input, outputs=predictions)

[ ] # Compile the model
model.compile(
    optimizer=Adam(learning_rate=0.001),
    loss="categorical_crossentropy",
    metrics=["accuracy"],
)

[ ] # Train the model with frozen base layers
initial_epochs = 15
history = model.fit(
    train_dataset,
    validation_data=val_dataset,
    epochs=initial_epochs,
)
```

Activate Windows
Go to Settings to activate Windows.

7s completed at 12:11 AM

FIGURE 7 DATA PREPROCESSING

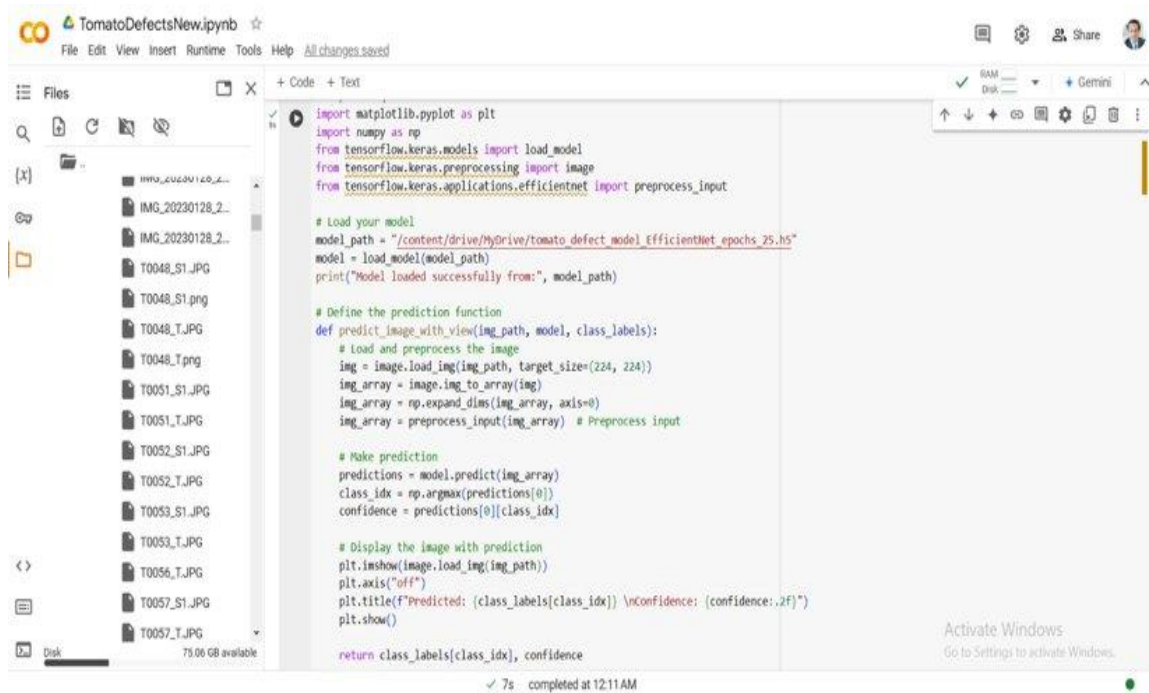


FIGURE 8 DATA PREPROCESSING

2.1.4. Model Development

2.1.4.1. Defect Detection with EfficientNet

The EfficientNet-B0 convolutional neural network architecture, which is renowned for its great accuracy and processing efficiency, is used in the defect detection component. Initial Model: Initially trained on the ImageNet dataset, the EfficientNet-B0 model was pretrained. To fine-tune the model to the particular defect classes, the top layers were swapped out for bespoke dense layers and tested on the tomato defect dataset. Improving the Architecture: To minimize dimensions, use the Global Average Pooling Layer. Completely Linked Dense Layers Activated by ReLU Multiple-class categorization using the Softmax Output Layer (cracks, rot, shriveling, and healthy) Loss Function: To assess multi-class classification loss, categorical cross-entropy was employed. Optimizer: In order to prevent local minima and speed up convergence, the Adam optimizer was used in conjunction with an adaptive learning rate schedule. Regularization: To avoid overfitting, L2 regularization and early ending callbacks were employed.

2.1.4.2. Quality Grading with MobileNetV2

A lightweight CNN that is perfect for use in mobile and Internet of Things settings, MobileNetV2 is used in the grading module. Learning via Transfer: For the grading job, the base MobileNetV2 model was modified and loaded with pretrained weights from ImageNet. The last layer of classification: Four neurons from a new dense layer (for Grade A, B, C, and Rejected) were added. To make predictions for several classes, softmax activation was used. The purpose of dropout layers was to avoid overfitting during training by introducing them in between dense layers. Training Improvements: Monitoring validation loss with early halting Improved convergence on a plateau with a lower learning rate The stability of training was improved via batch normalization and data shuffling. To prepare it for deployment, the model was transformed into TensorFlow Lite format for use in mobile apps. These development procedures made sure that both models were optimized for practical deployment in actual agricultural settings, as well as for accuracy and efficiency.

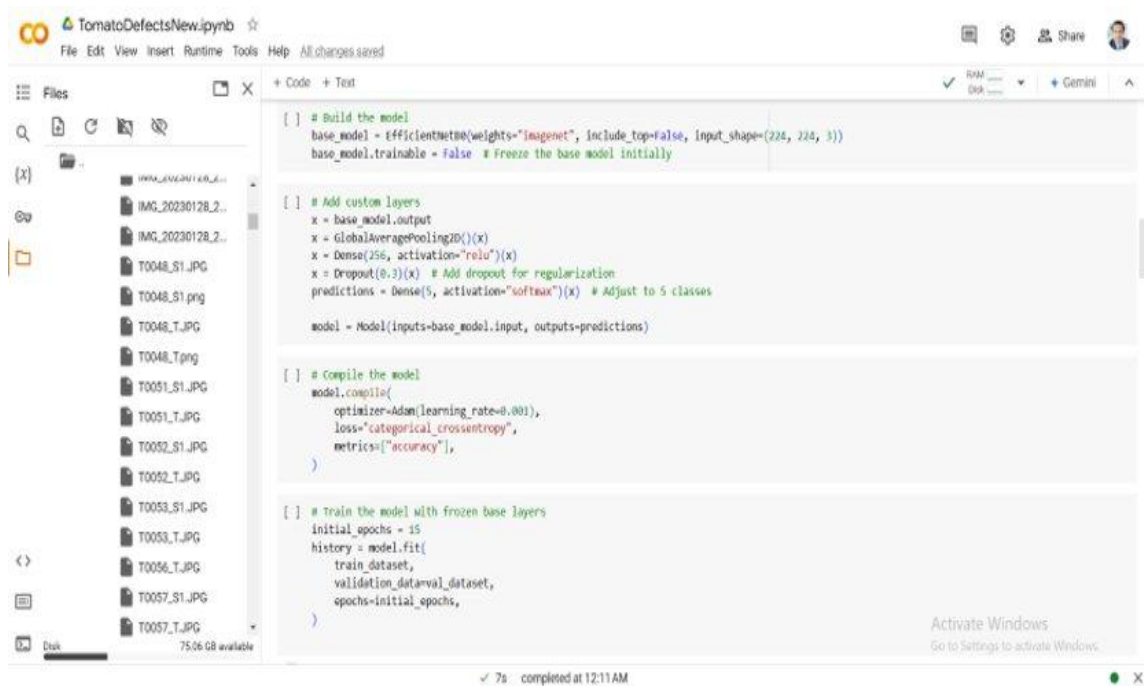


FIGURE 9 MODEL DEVELOPMENT

2.1.5. Model Training and Validation

A thorough training and validation procedure was used to guarantee the best possible performance from the fault detection and quality grading models. Split for Training and Validation: Two subsets of the dataset were randomly selected: 80% Training Set: Used to identify patterns in the data and fit the model parameters. 20% Validation Set: Useful for tracking generalization and identifying overfitting, this set evaluates the model's performance on unseen data during training. Settings for Training: 32 people in a batch 50 epochs (with early halting applied) Optimiser: Adam Learning Rate: Set at 0.001 initially, with a planned plateau decrease

Measures of Evaluation: Accuracy: Calculates the percentage of photos that are correctly classified. Precision: Shows the proportion of real positive results among all anticipated positive results. Recall (Sensitivity): Indicates how well the model can identify every pertinent instance. The F1-score, which is particularly helpful in datasets that are unbalanced, is the harmonic mean of precision and recall. Examining Validation:

Confusion Matrices: Produced for both models in order to visualize classification outcomes and spot misclassification trends. Error and Precision Curves: Plotted over time to assess patterns in model learning and identify the best places to stop. To confirm consistency across several dataset divisions and further validate model stability, cross-validation was also tested on a smaller scale. The models' ability to generalize effectively and function reliably under a variety of real-world tomato situations was guaranteed by this thorough training and validation approach.

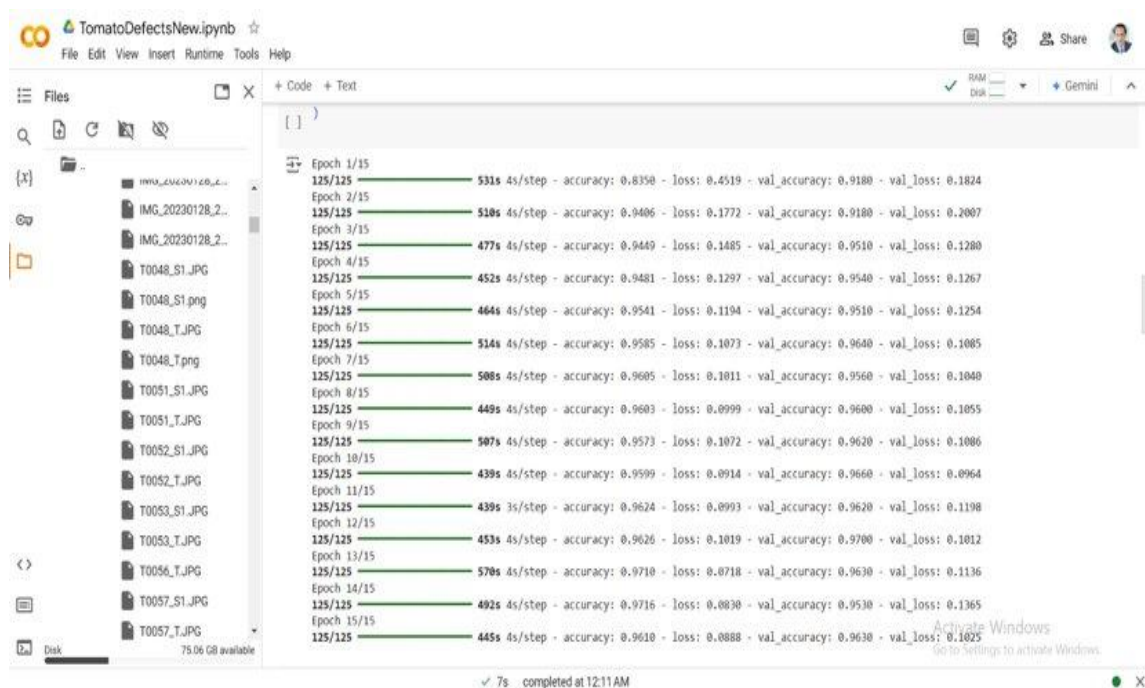


FIGURE 10 MODEL TRAINING AND VALIDATION

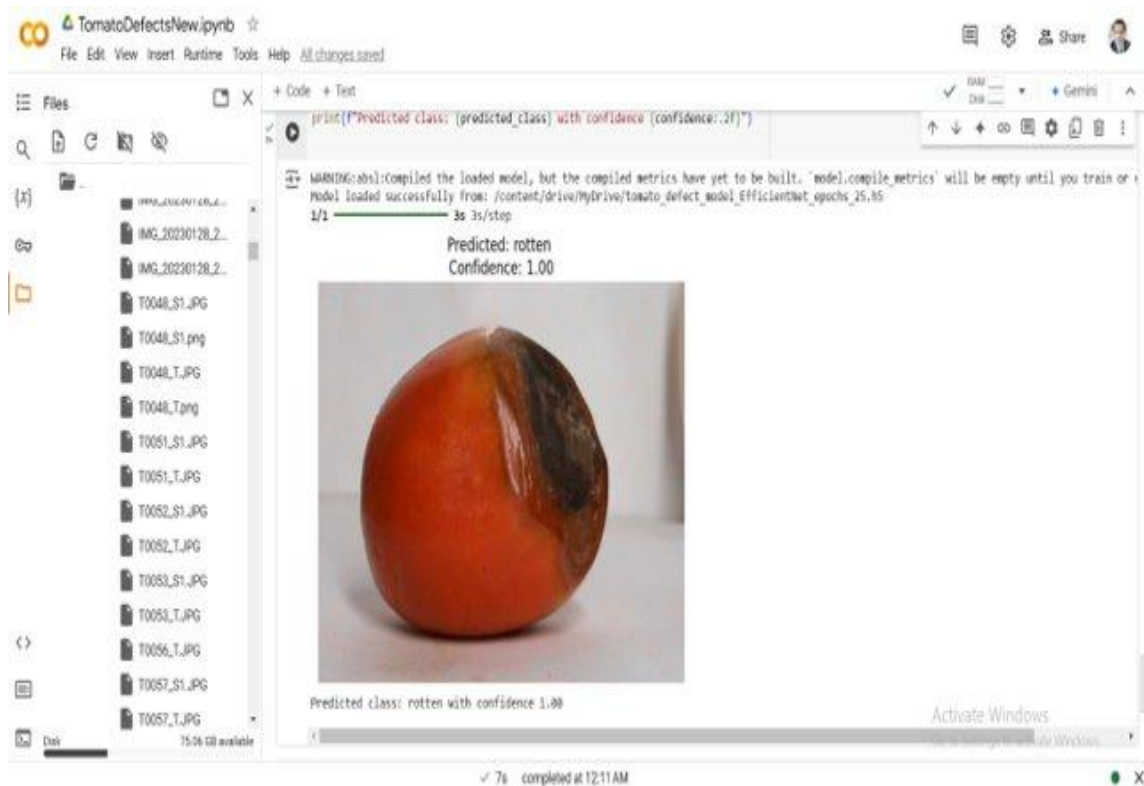


FIGURE 11 MODEL TRAINING AND VALIDATION

2.1.6. System Integration

The goal of the system integration phase was to integrate all of the model's components into a single operational pipeline that could be used in real time. Maintaining quick and dependable communication with the user interface while ensuring smooth interaction between the flaw detection and grading models was the aim. A unified pipeline for artificial intelligence The EfficientNet and MobileNetV2 models are both integrated into the system. The EfficientNet model is initially used by the system to evaluate flaws when an image is input. For quality grading, the output is sent to MobileNetV2 if the image is deemed to be defect-free or just slightly impacted. The outcomes of this two-step classification process are accurate and significant.

Backend API Services: The model inference engine and frontend data flow was managed via custom RESTful APIs created with Flask and FastAPI. These APIs consist of: Accept input images through HTTP requests Infer in real time using TensorFlow models that have already been loaded. Include prediction labels and confidence scores in the returned JSON

responses. Integrated Database: The ability to store prediction history and image logs in a MongoDB or Firebase database for later analysis, model enhancement, or auditing is available. Frontend Over Platforms: A user-friendly and responsive interface was created with the help of: For Android and iOS mobile users, Flutter: Facilitates offline image analysis and real-time forecasts For online users like agricultural researchers and warehouse operators, React.js Visualization of Results and Image Upload: Users have the option to either upload images or take real-time pictures with their device's camera. What the interface shows: Defect type found (if any) Graded for quality (A, B, C, Rejected) Scores for confidence and visual labels Performance optimization was achieved by loading the models into memory to shorten the inference time. Speed was improved by using asynchronous API queries and caching techniques. A completely automated, dependable, and easily accessible quality evaluation system that can function well in a variety of agricultural settings is guaranteed by this integration.

```

1 from flask import Flask, request, jsonify
2 from flask_cors import CORS
3 import tensorflow as tf
4 import numpy as np
5 import cv2
6 from PIL import Image
7 import os
8 from datetime import datetime
9 from tensorflow.keras.models import load_model
10
11 # Disable TensorFlow warnings
12 os.environ["TF_ENABLE_ONEDNN_OPTS"] = "0"
13
14 # Initialize Flask app
15 app = Flask(__name__)
16 CORS(app) # Enable CORS for frontend communication
17
18 # Load models safely
19 try:
20     efficientnet_model = load_model('tomato_defect_model_EfficientNet.h5', compile=False)
21     mobilenetv2_model = load_model('Quality_Grading_MobileNetV2.h5', compile=False)
22 except Exception as e:
23     raise ValueError(f"Error loading models: {str(e)}")
24
25 def get_quality_grade(quality_score):
26     """Map quality score to grade."""
27     if quality_score >= 90:
28         return "Grade A"
29     elif quality_score >= 75:
30         return "Grade B"
31     return "Grade C"
32
33 @app.route("/predict", methods=["POST"])
34 def predict():
35     if "file" not in request.files:
36         return jsonify({"error": "No file provided"}), 400
37

```

FIGURE 12 SYSTEM INTEGRATION

2.1.7. Interface and Usability Design

To assure usability for a broad spectrum of users, especially farmers and agricultural operators who might not have much experience with digital systems, the user interface was built with a heavy emphasis on simplicity, responsiveness, and accessibility. Capturing and uploading

images: The interface allows for both camera integration and image uploading. Using the camera on their mobile, users can take live tomato photos or choose pre-existing photos from local storage. This adaptability supports many user contexts, ranging from warehouse operations to field-level use. Feedback on Real-Time Predictions: The backend AI pipeline instantly analyzes an image after it is submitted and provides a prediction. What the user interface shows: Defect type found (if any) Given a quality rating (A, B, C, or rejected) Scores of confidence that show the certainty of the model

Visual Indicators: To improve interpretability, symbols, color coding, and unambiguous visual labels are included with the results. As an example: Green indicates superior quality, or Grade A Yellow denotes average quality, or Grades B and C. Red for serious or rejected flaws Experience for Users (UX): Made with the fewest possible navigational steps in mind Multiple language support for increased accessibility Font and button dimensions that are suited for mobile usability The ability to operate offline: Even without network connectivity, farmers can obtain forecasts thanks to the mobile version (developed with Flutter) that allows local model inference using TensorFlow Lite in rural areas with spotty internet. User feedback and interaction: Users can flag problems or provide comments on forecast accuracy, which are saved for use in future iterations of the model. This interface design makes high-tech tomato quality assessment simple, quick, and farmer-friendly by bridging the gap between sophisticated AI algorithms and actual agricultural users.

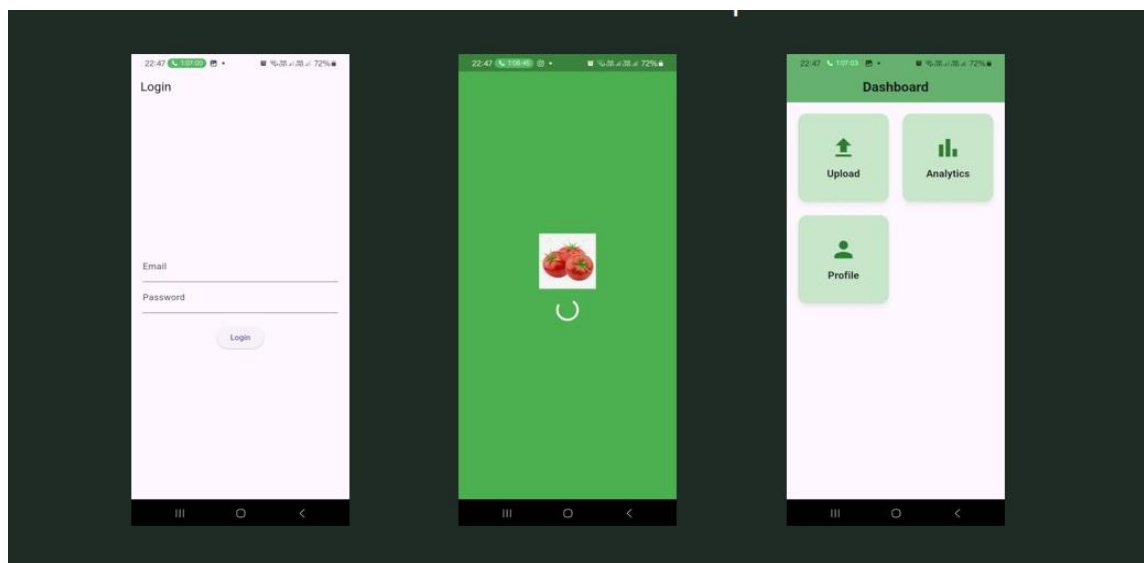


FIGURE 13 INTERFACE AND USABILITY DESIGN

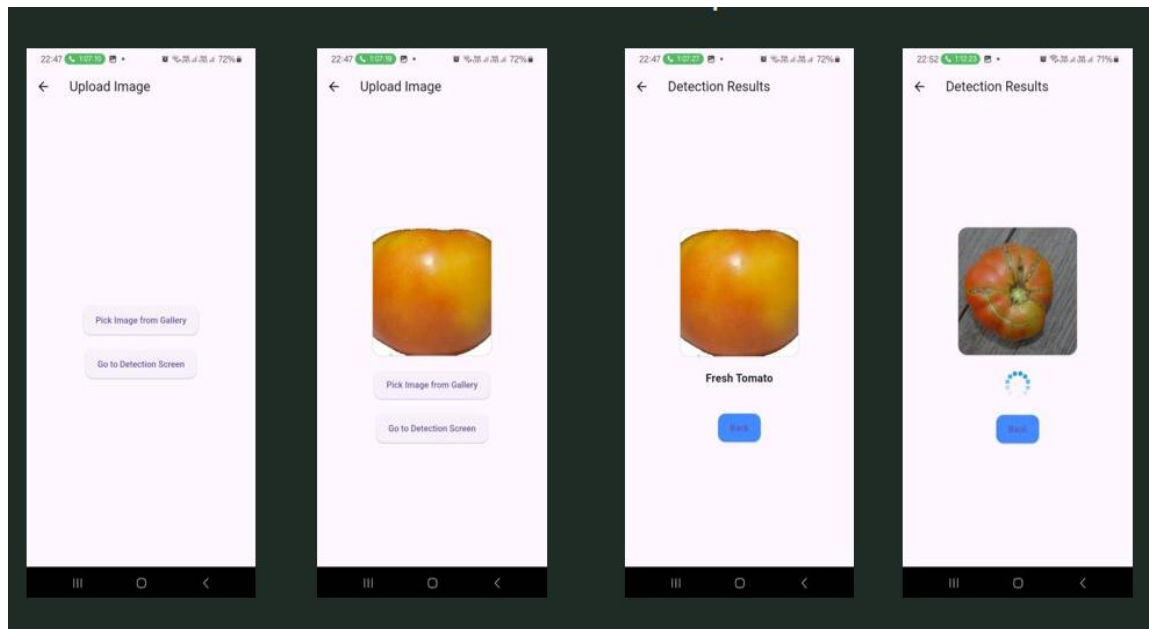


FIGURE 14 INTERFACE AND USABILITY DESIGN

2.1.8. Deployment

The finished system can be used in edge computing and mobile environments because it was designed with deployment flexibility in mind. This guarantees that people with different levels of technological infrastructure may access the content. Installing TensorFlow Lite on Mobile Devices: The TensorFlow Lite format was used to provide real-time inference on Android-based mobile devices using the trained EfficientNet and MobileNetV2 models. Users may rapidly do flaw identification and quality rating by taking or submitting tomato photos thanks to the smartphone app, which was developed using Flutter. With the low-latency performance guaranteed by mobile inference, predictions can be produced in less than two seconds. It was made possible to utilize it offline in places with inadequate or nonexistent internet access.

Deployment of Edge Devices (Raspberry Pi / Jetson Nano): The nVIDIA Jetson Nano and Raspberry Pi 4B, two reasonably priced and portable edge devices appropriate for use in agricultural fields, were used to test the system. Model changes were made to lower computational

cost and memory usage, guaranteeing seamless operation on low-resource devices. For image processing and inference on edge devices, OpenCV and the lightweight Flask API, which is based on Python, were utilized. For on-site data collection, devices were equipped with USB or Pi-compatible cameras, enabling real-time forecasting in far-flung farming settings. Support for Cross-Platforms: Docker was used to containerize backend services for flexible deployment in edge, cloud, and mobile settings. To ensure the safe handling of data, security features including secure API endpoints and encrypted image upload were introduced. The program will be scalable, responsive, and useable in a range of operational contexts, from industrial-scale grading facilities to smallholder farms with limited resources, thanks to this deployment technique.

```

33 @app.route("/predict", methods=["POST"])
34 def predict():
35     if "file" not in request.files:
36         return jsonify({"error": "No file provided"}), 400
37
38     file = request.files["file"]
39     if file.filename == "":
40         return jsonify({"error": "No file selected"}), 400
41
42     try:
43         file_bytes = np.frombuffer(file.read(), np.uint8)
44         img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
45         if img is None:
46             raise ValueError("Invalid image format")
47
48         defect_result = process_defect_detection(img)
49         quality_result = process_quality_grading(img)
50
51         return jsonify({
52             "defect_detection": defect_result,
53             "quality_grading": quality_result,
54             "timestamp": datetime.now().isoformat()
55         })
56     except Exception as e:
57         return jsonify({"error": f"Image processing error: {str(e)}"}), 500
58
59
60 def preprocess_image(img, target_size=(224, 224)):
61     """Resize and normalize image for TensorFlow models."""
62     img_pil = image.fromarray(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
63     img_pil = img_pil.resize(target_size)
64     img_array = np.array(img_pil) / 255.0 # Normalize
65     return np.expand_dims(img_array, axis=0)
66
67 def process_defect_detection(img):
68     """Defect detection using EfficientNet."""

```

FIGURE 15 INTERFACE AND USABILITY DESIGN

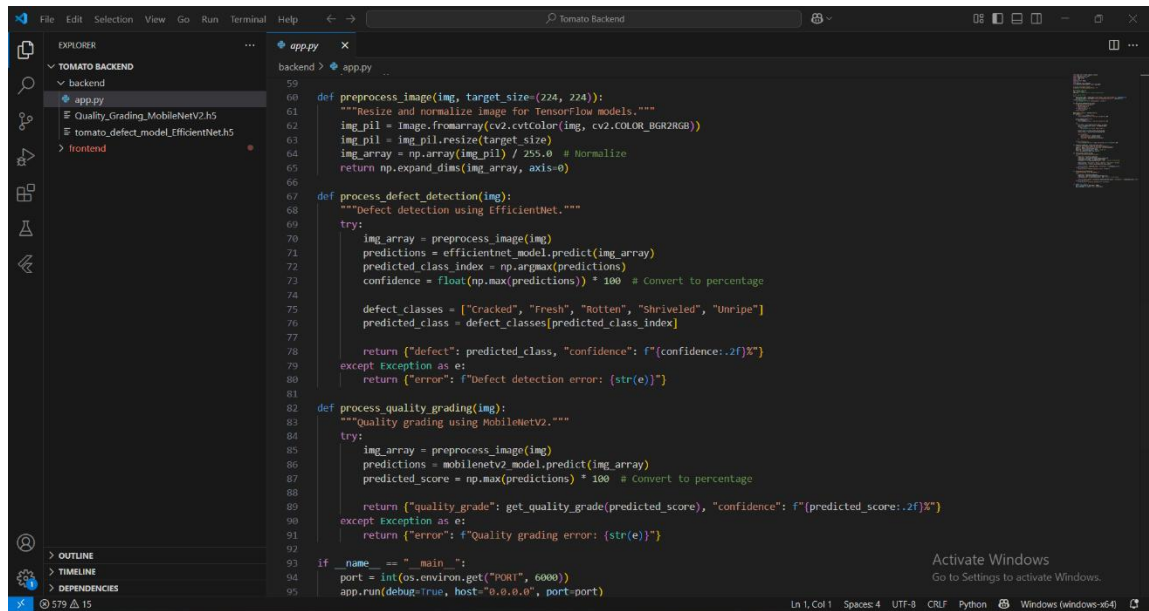


FIGURE 16 DEPLOYMENT

2.2. Commercialization Aspects of the Product

A number of industry-wide issues are addressed by this AI-powered tomato defect detection and quality grading system: Lack of reasonably priced and trustworthy quality assessment solutions; reliance on manual inspection, which is frequently unreliable; and high rejection rates at export checkpoints because of undetected defects. The product is made to satisfy the practical needs of different stakeholders in agriculture, including smallholder farmers, distributors, exporters, and the food processing industries. The commercialization strategy emphasizes accessibility, affordability, scalability, and real-world usability.

For farms and packing facilities, the solution can be sold as a hardware-integrated package in addition to a stand-alone smartphone app. Because to its modular design, it can be used by both small farmers and major agricultural enterprises at a variety of operational scales. The commercialization road also entails collaborations with NGOs engaged in smart farming projects and government agriculture ministries. Long-term adoption, training, and distribution will all benefit from these partnerships.

2.2.1. Target Market

A wide range of agricultural and food supply chain industries are the target market for our AI-powered tomato quality evaluation system. Among these are: Local Farmers: Smallholder and medium-sized farmers want reasonably priced, user-friendly equipment to evaluate and grade their produce prior to market distribution. With the help of the mobile-based solution, they may decide on harvesting, pricing, and selling with knowledge. Organizations that export agricultural products must adhere to stringent quality standards. By limiting rejection rates and ensuring that only tomatoes of superior quality and free of defects are chosen for shipping, the approach helps to increase international trust.

Food Processing Centers: The method can be used to automate grading in factories and processing facilities that turn tomatoes into value-added products (purées, canned tomatoes, and sauces), increasing batch consistency and cutting labor expenses. Government Agricultural Departments: Organizations that work on rural development and quality assurance can use the system for field testing, training initiatives, and encouraging farmers to go digital. National food quality monitoring programs might also incorporate it. Retail Chains and Wholesalers: By automating quality inspections at collection hubs, the technology helps supermarkets and distributors maintain quality standards across supply batches. The system can be implemented with customized price and feature models in each of these market categories, allowing for longer-term sustainability and wider market penetration.

2.2.2. Business Model

A wide spectrum of users with different financial and technological capacities can be served by the business model. Flexibility, affordability, and scalability access are guaranteed by a tiered model: The Free Tier is intended for rural residents and smallholder farmers. includes fundamental features including the ability to detect defects in single images. Pre-installed TensorFlow Lite models are used for offline prediction via mobile devices. Perfect to raise awareness and educate people.

Premium Tier: Targeted at food processing industries, exporters, and agribusinesses. offers improved features, such as bulk analysis and batch picture upload Analytical prediction and past grades Priority assistance and system configuration Dashboard in the cloud for tracking and

exporting data Support for integration with external logistics and inventory systems This paradigm facilitates the creation of income while encouraging diversity and providing all tiers of agricultural stakeholders with access to contemporary technology.

2.2.3. Monetization Plan

In order to provide value to various user groups and maintain financial sustainability, a well-structured monetization strategy is essential. Proposed revenue streams include the following: Monthly/Yearly Subscription for Mobile Apps: Users can choose from a variety of options according to their usage requirements. Core functions and a monthly restriction on scans are included in the basic plan. Standard Plan: Batch uploads, unlimited scans, and grading analytics are all included. Advanced capabilities like API access, warehouse system integration, and customer assistance are all part of the enterprise plan, which is designed specifically for agribusinesses.

Third-Party Integration API Access: Agricultural research institutions, logistics companies, and agritech platforms can all incorporate the fundamental AI models into their systems using API endpoints. A license-based model or pay-per-use pricing is used to provide access. Services for Consulting and Custom Deployment: Provide white-label solutions, model fine-tuning, or customisation to government programs, cooperatives, and exporters. consists of training local field officers and installing edge-based equipment. CSR Partnerships and Institutional Partnerships: Collaborate with rural development initiatives, agricultural universities, and non-governmental organizations to support deployments. The system can be implemented in underprivileged areas at a reduced or no cost by utilizing grants and CSR monies. By promoting both social impact and economic scale-up, this multi-channel monetization strategy facilitates broad acceptance and ongoing innovation.

2.2.4. Competitive Advantage

The system's inventiveness, flexibility, and applicability to the actual agricultural ecosystem provide it a competitive advantage. The following characteristics set it apart in the expanding agri-tech solutions market: Both offline and real-time functionality are available, providing almost immediate grading and defect predictions. Rural and isolated locations with inadequate

connectivity can benefit from offline model inference with TensorFlow Lite. Compliance with Mobile and Edge Devices: Effectively operates on Jetson Nano, Raspberry Pi, and smartphones. allows for field and post-harvest center use without requiring costly hardware or cloud infrastructure. Region-Specific Training Data: Photos gathered from nearby farms are used to refine the models, representing actual defect kinds and quality classifications unique to the area. This improves the model's relevance and accuracy in comparison to generic, globally trained systems.

Scalability Across Markets: Easily retrains or fine-tunes using new datasets to accommodate different crops and geographical areas. Useful in foreign and domestic markets, such as quality checking facilities, farmer cooperatives, and export chains. Accessibility and User-Centered Design: Feedback from real farmers and supply chain managers informed the design. An intuitive and multilingual interface promotes adoption among many user groups. These characteristics give the system a reputation as a dependable, scalable, and easily accessible instrument that connects AI innovation with the real-world requirements of agriculture.

2.2.5. Future Scope

Using artificial intelligence to revolutionize digital agriculture and post-harvest management, this project's long-term goals go beyond tomatoes. Potential advancements and prospects in the future comprise: Spread to Additional Crops: Incorporate additional perishable food, such as bananas, mangoes, chiles, and green vegetables, into the system to facilitate defect identification and quality grading. For effective adaptation of the present architecture to new crop datasets, use transfer learning and modular retraining techniques. Support in Multiple Languages: Multilingual user interfaces that are adapted to native languages and regional dialects should be introduced to improve accessibility in rural farming areas. Assist customers with low reading levels by offering text-to-speech capabilities and audio-based coaching.

Agricultural supply chain management systems and e-marketplaces can be integrated with the app to enable smooth post-grading commerce. To improve farm-to-market traceability and pricing information, enable batch tracking and automatic tagging. Using picture data and growth patterns, AI-Driven Yield Prediction and Advisory uses predictive models to anticipate crop health, expected yield, and harvest dates. Make suggestions on the best circumstances for harvesting, packing, and transportation. Analytics Dashboard in the Cloud: Create a centralized analytics site that allows users, researchers, and agribusinesses to assess defect incidence over time, compare

quality scores by region, and track historical patterns. Model Updating and Collaborative Learning: To continuously enhance the model utilizing data from real-time users without jeopardizing data privacy, employ edge model retraining or federated learning. These upcoming improvements are intended to improve the platform's value offer, encourage sustainable farming methods, and provide stakeholders with tools for wise decision-making throughout the value chain.

2.3. Testing and Implementation

2.3.1. Testing Phases

For the system to be accurate, reliable, and resilient in the actual world, testing was essential. To validate the system as a whole and its separate components, a variety of testing techniques were used. To verify proper functionality, unit testing was done on each model separately (MobileNetV2 for quality grading and EfficientNet for defect detection). We verified learning behavior by comparing TensorFlow model predictions with predicted labels. To ensure consistency, tests were conducted on input reshaping routines, augmentation logic, and preprocessing pipelines.

Integration Testing: The components for grading and flaw detection were tested as a single, integrated system. Correct input/output handling, including edge cases and numerous concurrent queries, was verified in the backend API answers. Frontend (Flutter/React) and backend (Flask/FastAPI) communication flow was verified for several use-case scenarios, including batch uploads and single picture processing. Acceptance testing by users (UAT): conducted field tests involving more than ten actual users, such as agricultural officers, farmers, and quality control managers. Through questionnaires and in-person demos, input was gathered. Evaluation was centered on output interpretability, responsiveness, usability, and interface clarity. Every testing phase improved the system's scalability, accuracy, and usability, guaranteeing its preparedness for practical agricultural applications.

2.3.2. Evaluation Metrics

The effectiveness of the quality grading and defect detection models, as well as the system's responsiveness and suitability for real-time use, were all validated with the use of evaluation metrics. The measures listed below were employed: By comparing the number of accurate forecasts to the total number of predictions, accuracy assesses the model's overall correctness. It offers a high-level summary of the model's performance, which is especially helpful in balanced classes.

Precision is a measure of how many accurately anticipated positive observations there are in relation to all predicted positives. In situations where the cost of false positives is considerable, it is vital. Recall evaluates a model's capacity to locate every pertinent instance in a dataset. It is crucial when omitting a faulty product could lead to problems with quality later on. For a fair assessment, these metrics were computed for every class (crack, rot, Grade A, B, etc.).

An impartial statistic that takes into account both false positives and false negatives is the F1-Score, which is the harmonic mean of precision and recall. Particularly helpful with managing unbalanced datasets, which are typical in practical agricultural applications. Latency: Calculated how long it took to display the prediction result once an image was submitted. According to benchmarks for mobile and edge devices, average latency stayed below two seconds, guaranteeing appropriateness for field use in real time. The accuracy, sensitivity, and viability of the system for implementation in dynamic, real-world agricultural environments were evaluated using these criteria taken together.

F1-Score: A balanced statistic that accounts for both false positives and false negatives, it is the harmonic mean of precision and recall. excellent for managing unbalanced datasets, which are prevalent in practical agricultural applications. Latency: The amount of time between submitting an image and seeing the prediction result was measured. Benchmarks for mobile and edge devices revealed that average latency stayed below 2 seconds, guaranteeing appropriateness for field use in real time. All of these criteria worked together to evaluate the system's sensitivity, accuracy, and suitability for use in dynamic, real-world agricultural environments.

2.3.3. Real-World Testing

Extensive real-world testing was carried out in various agricultural situations to guarantee that the designed system functions well under actual farming and operational circumstances. Taking

Pictures in Natural Environments: Images of tomatoes were gathered straight from outdoor marketplaces, packaging facilities, and farms. Examples with different lighting conditions, background clutter, angles, and actual flaws were included in the images. This made it easier to model the erratic and unmanaged visual conditions that are typical in the field. **Mobile Device Assessment:** In order to assess performance across hardware tiers, the system was tested on a number of mobile devices with varying specs. To guarantee continuous responsiveness and usability, devices such as tablets, mid-range models, and inexpensive Android smartphones were tested.

Assessing Field Conditions: Using edge devices like Raspberry Pis and Jetson Nanos linked to USB cameras, on-site assessments were carried out in greenhouses and open farms. Through these tests, the deployed systems' power consumption, camera compatibility, and environmental resilience were evaluated. Usability observations were made when field workers and farmers were utilizing the mobile application. Feedback was obtained on the user interface's responsiveness, predictability results' clarity, and simplicity of navigation. **Across Image Quality Reliability:** Predictions were tested for robustness on low-resolution, somewhat blurred, and overexposed/underexposed photographs. The models' high accuracy was maintained under a variety of image circumstances, confirming the efficacy of the preprocessing procedure. The system's preparedness for large-scale implementation was demonstrated by this thorough real-world testing, which confirmed that it produces accurate, dependable, and interpretable data in uncontrolled agricultural settings.

2.3.4. Feedback & Iteration

The program was refined based in large part on user feedback to guarantee functionality and usability in the actual world. Interviews, organized field trials, and usability workshops with a wide range of users were used to get this input. **Getting Input:** Around ten end users, including exporters, agricultural field officers, and smallholder farmers, participated in sessions. In order to fill out feedback forms and provide verbal thoughts, participants utilized the program on mobile devices in authentic farm settings. Live encounters were also observed in order to record nonverbal usability indicators.

Improvement Needs: Users pointed out that color-coded grading indicators and more lucid output labels were necessary. It was suggested that the interface for uploading images be improved simpler and that visual instructions be included for higher-quality photography.

Faster load times for low-end devices, battery optimization, and offline result saving were among the recommendations. System Changes: The user interface (UI) was rebuilt with bigger buttons, more readable feedback icons, and improved accessibility features (including support for several languages). More varied tomato photos depicting uncommon defect cases and environmental variances were included to the dataset. The performance of the inference models was considerably improved, particularly on devices with lower specifications.

Ongoing Iteration Strategy: For ongoing reference and product planning, feedback logs were kept in a database. Using an agile development cycle, user-driven enhancements were ranked according to priority within sprints. By ensuring that the application changed to meet genuine user needs and operational realities, this iteration cycle and ongoing input increased the likelihood and effect of adoption.

2.3.5. Final Deployment

The system's ultimate deployment included both cloud-based and mobile components, making it scalable, usable, and accessible for farmers, quality inspectors, and agricultural organizations. Mobile Model Implementation: After being trained, the EfficientNet and MobileNetV2 models were exported to TensorFlow Lite and integrated into a mobile application built with Flutter. The application underwent extensive testing on Android smartphones and was tuned to perform inference locally, allowing for predictions without the need for internet access. It allowed users to take pictures of tomatoes, get real-time findings, and save predictions for later review.

Cloud and API Deployment: A small, lightweight Flask API that can submit images remotely and return predictions in JSON format was created and put live on Heroku. This facilitated future growth to cloud-based dashboards, remote access through web portals, and connection with third-party platforms. The API facilitates scalable load balancing for numerous users as well as safe picture handling. Pilot Testing Distribution of Android Apps: Google Play Beta Testing and APK files were used to deploy the program to test users. Final improvements to the UI/UX and backend response processing were influenced by feedback from pilot testing.

Training and Documentation: To assist farmers and field officers, a user manual was created that included guidance on installation, image capture methods, and result interpretation. Training workshops and outreach events included live demonstrations and instructional films. The successful transition from prototype to real-world application was made possible by this multifaceted deployment strategy, opening the door for additional scale-up and commercial launch.

3. RESULTS AND DISCUSSION

3.1. Results

In both the tomato flaw detection and quality rating tasks, the final AI system performed remarkably well, fulfilling the project objectives. Through a thorough series of experiments that comprised both structured validation datasets and uncontrolled real-world field circumstances, the trained models were assessed. This two-pronged testing strategy guaranteed both practical dependability and empirical robustness. The models were evaluated during the validation phase using statistical measures like accuracy, precision, recall, and F1-score, which verified that they could effectively generalize to new data. Multiple scenarios with varying lighting, backdrop, and device circumstances were used to test the system in field deployments. These practical tests confirmed the model's responsiveness and flexibility.

The models' effective operation shows that they are prepared for implementation in agricultural supply networks. To ensure that it is suitable for farmers' and field officers' real-time use, system latency was tested on mobile and edge devices in addition to classification accuracy. The comprehensive quantitative results of both model evaluations are shown in the sections that follow.

3.1.1. Defect Detection (EfficientNet)

TABLE 3 EFFICIENTNET MODEL PERFORMANCE COMPARISON

Metric	EfficientNet (Defect Detection)
Precision	93.2%
Average precision	91.4%.
Average recall	90.1%.
Average F1-score	90.7%

When it came to different tomato fault classes, the EfficientNet model showed excellent classification ability. It was especially good at differentiating between samples that were clearly flawed and those that were healthy. Due to their unique visual characteristics, such as black

blotches and surface fissures, rot and cracks were identified with the highest consistency, according to the confusion matrix study. Although less common in the sample, shrimpling occasionally produced false negative results because of its faint resemblance to typical tomato wrinkling patterns. This was particularly noticeable in samples taken in strong or erratic illumination.

The model's balanced performance in terms of precision and recall, which minimizes false positives and false negatives, is further supported by the strong F1-score. The model demonstrated resilience to changes in background and camera angle during field tests, preserving a steady forecast accuracy. To sum up, EfficientNet-B0 demonstrated itself to be a viable architecture for the real-time categorization of tomato defects, providing a stable basis for automated quality assessment systems implemented on edge and mobile platforms.

3.1.2. **Quality Grading (MobileNetV2)**

TABLE 4 MOBILENETV2 MODEL PERFORMANCE COMPARISON

Metric	MobileNetV2 (Quality Grading)
Accuracy	90.5%
Average precision	88.9%
Average recall	89.3%
Average F1-Score	89.1%

When grading tomatoes into the predetermined commercial categories of Grade A, B, C, and Rejected, the MobileNetV2 model demonstrated dependable performance. High classification consistency was attained by the model on both real-world image samples and a variety of test sets. The highest level of confidence was used to classify rejected and grade A tomatoes. These grades had more recognizable visual characteristics, such as Grade A tomatoes' consistent color and shape and Rejected tomatoes' severe rot or malformations. There was less misclassification in these categories as a result of this uniqueness.

However, there was more overlap in grades B and C. The main causes of misclassifications between these two nearby grades were: Tiny changes in size and color minor surface flaws that, depending on the situation, could be classified as either grade Subjectivity of human labeling during data annotation The F1-score showed that the overall performance was still strong in spite of these overlaps. According to the confusion matrix, the majority of misclassifications still happened within limits of acceptable commercial tolerance. The model's ability to retain grading accuracy in the presence of natural sunlight and uneven backdrops was validated by field testing. This demonstrates MobileNetV2's resilience for real-time, lightweight deployment in supply chain and farm-level settings. These outcomes demonstrate that MobileNetV2 may function as a dependable core engine in automated tomato grading systems that enhance uniformity, impartiality, and quality control efficiency.

3.1.3. Latency Performance

The average latency for mobile devices is 1.4 seconds. 2.1 seconds for edge devices (Raspberry Pi and Jetson Nano). For systems built for real-time deployment, latency—the amount of time between taking an image or uploading it and getting the prediction output—is an essential performance metric. In order to minimize inference delay without sacrificing accuracy, the AI pipeline was tuned. TensorFlow Lite-optimized models produced predictions in less than 1.5 seconds on contemporary Android handsets, guaranteeing responsiveness appropriate for farm-level applications. The average end-to-end latency, including preprocessing and model execution, was about 2.1 seconds, even on resource-constrained edge devices like the Raspberry Pi and Jetson Nano.

This performance confirms that the system is appropriate for on-site, real-time applications where quick response is crucial, such harvesting, sorting, or quality checks at collection centers. Batch analysis processes are supported by the low latency, which enables the sequential processing of several tomatoes with little time overhead. Furthermore, this responsiveness held true across a variety of device configurations and environmental circumstances, highlighting the deployed solution's efficiency and mobility. For field-based agricultural decision-making and logistical activities, these latency metrics fall well inside the acceptable range.

3.2. Research Findings

The following noteworthy results were obtained from the project, all of which support the validity and possible practical implementation of the suggested remedy: **Model Effectiveness:** Deep learning models that strike a good mix between computational efficiency and prediction accuracy, such as EfficientNet and MobileNetV2, are ideal for visual quality classification tasks in agriculture.

The Value of Data by Region: Models' capacity to generalize in actual agricultural settings was much improved by training them with photos from nearby farms and markets. This highlighted how important localized datasets are to the development of agri-tech AI. **User-Centered Design Is Important:** Large icons, color-coded outputs, easier navigation, and support for local languages were among the features that significantly increased user acceptance, particularly among farmers with little technical experience.

Offline and Edge Capability: TensorFlow Lite's offline and edge device functionality was crucial, particularly in remote locations with spotty or nonexistent internet access. **Benefits of Feedback Loops** The model and interface were regularly improved by incorporating input from actual users, such as farmers, exporters, and agricultural officers. This helped to match the system's capabilities with the needs of the field and realistic user expectations. **Scalability and Adaptability:** Beyond tomatoes, the architecture can be fine-tuned or retrained for various crops. A multi-crop intelligent grading platform is thus established. All of these studies show how AI-powered mobile and edge

solutions have the potential to transform post-harvest quality control, making it more accurate, quicker, and available to the larger agricultural community.

3.3. Discussion

The outcomes show how well the suggested AI-based tomato evaluation system automates and improves the process of grading and identifying flaws. In both lab-based validation and uncontrolled field settings, the dual-model architecture—which consists of MobileNetV2 for quality grading and EfficientNet for defect detection—repeatedly produced good accuracy and interpretability. This demonstrates that the system can manage the unpredictability of the real world while still generating dependable results. This solution's real-time predictive capabilities, which is enabled by its deployment on mobile and edge platforms, is one of its main advantages. The method overcomes a significant obstacle in rural agriculture technology adoption by utilizing TensorFlow Lite and lightweight deep learning architectures to deliver immediate feedback without requiring continuous internet access.

Incorporating user-centered design elements, like offline functionality, language support, and simpler interfaces, also had a big impact on user acceptability. Because it can be used by both technically proficient operators and farmers with low levels of digital literacy, the system is widely usable and accessible. Another significant benefit that came to light was the system's scalability. It is possible to expand the system to other crops or food items with comparable grading requirements because the existing design allows for retraining or transfer learning. Future applications in more general supply chain, food safety, and agricultural contexts are made possible by this.

3.3.1. Limitations

Although the results were encouraging, some restrictions were noted: Image Quality Sensitivity: Images with poor lighting or blur were found to perform worse, which may cause misclassification. Grade Ambiguity: Because of the subjectivity of the distinction between Grade B and C, there was considerable misunderstanding. Adding more varied samples to the training dataset or improving the grading standards might be necessary to achieve this. Restricted Dataset for Rare Cases: Despite the use of data augmentation, the training set contained a disproportionately small number of extremely rare defect cases, which may have an impact on the model's ability to generalize in edge circumstances.

3.3.2. Future Opportunities

Several tactics are suggested to address these issues and enhance the system even more: utilizing auto-correction or adaptive picture enhancing methods to make up for sharpness and lighting issues. To enable real-time learning from user-corrected predictions, an active learning module is being developed. adding more samples to the collection from various tomato kinds, seasons, and geographical areas. The prototype system has, in summary, shown the viability and significance of using AI into tomato quality grading. With continued improvement, cooperation from stakeholders, and strategic implementation, this system might completely transform agricultural quality management, increasing supply chain transparency, lowering losses, and promoting equity.

The findings show that the suggested AI-based tomato evaluation system efficiently improves and automates the process of grading and identifying flaws. The accuracy and interpretability of the dual-model design were consistently demonstrated in both laboratory and field settings. The system's integration of offline support, edge deployment, and real-time prediction made it extremely suitable for rural and under-resourced settings. Furthermore, the model can be retrained to accommodate different crops or even industrial use cases, according to its scalability. The following were among the limitations: minimal performance deterioration for photos with very bad lighting or low resolution. The ambiguity between Grade B and C categories may necessitate the integration of subjective comments or extra data.

Notwithstanding these drawbacks, the prototype's outcomes provide compelling evidence of the potential of AI-assisted agricultural quality grading. This technology may greatly increase the tomato supply chain's transparency, effectiveness, and profitability with more testing, selective marketing, and ongoing iteration.

4. Conclusions

The creation and implementation of an AI-powered system for quality grading and tomato fault detection has demonstrated how contemporary deep learning approaches may greatly enhance conventional methods for evaluating agricultural quality. By merging EfficientNet with MobileNetV2, the dual-model system replaced human judgment with reliable automated decision-making, offering extremely accurate, real-time insights about the physical condition and commercial value of tomatoes. Through extensive testing and practical validation, the system proved that it could function effectively on mobile and edge devices, making it usable even in farming contexts with limited resources or remote locations. The platform effectively handled issues with environmental variability, technological literacy, and limited infrastructure. Adoption among end users in rural communities was aided by crucial features including offline functionality, language support, and an easy-to-use interface.

The project's goals were met by: Using scalable and lightweight AI models to automate tomato flaw detection and grading. delivering low latency inference in real time on edge and mobile devices. making sure that a variety of device kinds and user profiles can be used and deployed. outperforming manual inspection techniques in quantifiable ways in terms of prediction accuracy, consistency, and usefulness. Additionally, the study pinpointed important prospects for advancement: The system may be fine-tuned and retrained to scale to various crops and perishable goods. Traceability and deeper insights may be provided by integration with supply chain platforms and cloud-based analytics dashboards. The use of feedback-based learning may facilitate ongoing development and adaptive model upgrades.

Notwithstanding the system's strong performance, its drawbacks—such as a minor deterioration in low-quality images and unpredictability in subjective grading, particularly between Grades B and C—indicate room for improvement. Incorporating sensor fusion (e.g., temperature, moisture), improving class borders, and growing the dataset are possible extensions. Conclusively, this study highlights the revolutionary possibilities of artificial intelligence within the agricultural sector. The suggested system gives farmers access to smart and easily available instruments while simultaneously enhancing quality control and lowering post-harvest losses. The future of smart farming and sustainable food production can be significantly shaped by this system with further funding, legislative backing, and stakeholder involvement.

The creation and implementation of an AI-powered system for quality grading and tomato fault detection has demonstrated how contemporary deep learning approaches may greatly enhance conventional methods for evaluating agricultural quality. Real-time, extremely precise insights into the physical state and commercial worth of tomatoes were obtained by the dual-model architecture that combined EfficientNet and MobileNetV2. Through extensive testing and practical validation, the system proved that it could function effectively on mobile and edge devices, making it usable even in farming contexts with limited resources or remote locations. In order to promote end-user acceptance, important features including offline functionality, language support, and an intuitive interface design were essential.

The project's goals were successfully met by: AI models for automated tomato flaw identification and grading Making sure that various hardware and user skill levels are usable and deployment-ready proving quantifiable gains over manual techniques in terms of accessibility, speed, and accuracy The study also found ways to expand the system to other crops and incorporate it into supply chain solutions and broader agricultural ecosystems. The foundation established by this study offers a strong platform for future improvements, even though certain restrictions like image quality sensitivity and grading subjectivity still exist.

In summary, this study demonstrates how artificial intelligence has the potential to revolutionize the agricultural industry. The approach created here has the potential to significantly improve food quality, cut waste, and raise the standard of living for farmers in a variety of geographical areas with additional funding, iteration, and cooperation.

5. Reference

- Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning (ICML), 97, 6105–6114. PMLR.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4510–4520.
- TensorFlow. (2023). TensorFlow Lite. Retrieved from <https://www.tensorflow.org/lite>
- Google Developers. (2023). Machine Learning Crash Course. Retrieved from <https://developers.google.com/machine-learning/crash-course>
- Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools. <https://opencv.org>
- Heroku. (2023). Cloud Application Platform. Retrieved from <https://www.heroku.com>
- Raspberry Pi Foundation. (2023). Raspberry Pi 4 Model B. Retrieved from <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>
- NVIDIA. (2023). Jetson Nano Developer Kit. Retrieved from <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>
- Chollet, F. (2017). Deep Learning with Python. Manning Publications.
- Brownlee, J. (2022). Machine Learning Mastery With Python: Understand Your Data, Create Accurate Models, and Work Projects End-To-End. Machine Learning Mastery.
- Computer Vision Based Fruit Grading System for Quality Evaluation of Tomato in Agriculture industry [<https://www.sciencedirect.com/science/article/pii/S1877050916001861>]
- Fruit-Based Tomato Grading System Using Features Fusion and Support Vector Machine [https://link.springer.com/chapter/10.1007/978-3-319-11310-4_35]
- A microcontroller based machine vision approach for tomato grading and sorting using SVM classifier

[\[https://www.sciencedirect.com/science/article/abs/pii/S0141933119307057\]](https://www.sciencedirect.com/science/article/abs/pii/S0141933119307057)

- A cost effective tomato maturity grading system using image processing for farmers [<https://ieeexplore.ieee.org/abstract/document/7019591>]

6. Glossary

AI (Artificial Intelligence): A field of computer science that enables machines to perform tasks that typically require human intelligence, such as learning, reasoning, and problem-solving.

CNN (Convolutional Neural Network): A deep learning algorithm particularly effective for processing grid-like data such as images, commonly used in visual recognition tasks.

EfficientNet: A family of convolutional neural networks developed by Google that scales depth, width, and resolution in a balanced way to achieve state-of-the-art performance with fewer parameters.

MobileNetV2: An optimized deep learning architecture for mobile and embedded vision applications that uses depthwise separable convolutions to reduce complexity and improve speed.

TensorFlow Lite: A lightweight version of TensorFlow designed for deploying machine learning models on mobile, embedded, and edge devices.

Latency: The time delay between initiating a command or input (e.g., uploading an image) and receiving a system response or output.

Precision, Recall, F1-Score:

Precision: The ratio of correctly predicted positive observations to the total predicted positives.

Recall: The ratio of correctly predicted positive observations to all actual positives.

F1-Score: The weighted average of Precision and Recall, used to balance the trade-off between the two in classification tasks.

Edge Devices: Computing hardware that performs data processing at or near the data source (e.g., Raspberry Pi, Jetson Nano), reducing the need for constant cloud connectivity.

Image Augmentation: A technique used in training machine learning models where images are modified through transformations such as rotation, flipping, and zooming to improve model generalization.

7. APPENDICES

Sample Images from the Dataset Appendix A This appendix contains a typical collection of tomato images that were utilized for both validation and training. Labels classifying the quality grade (Grades A, B, C, or Rejected) and fault kind (such as fractures, rot, or shriveling) are attached to each photograph. Real-world field settings and controlled test scenarios are both reflected in these annotated photos. Appendix B: Excerpts of Code contains the main Python code segments created for this project, such as: Resize, normalize, and supplement data as part of preparation procedures TensorFlow/Keras Flask API route handlers can be used to train scripts for the EfficientNet and MobileNetV2 models for image upload and prediction endpoint responses. Conversion code for TensorFlow Lite for mobile implementation User Feedback Form Template (Appendix C) a methodical survey form intended to gather input from farmers and other users while field testing is underway. The form contains the following: Questions about usability (such as results clarity and simplicity of navigating) Likert scale assessment of satisfaction Improvement suggestions Optional open-ended feedback area Appendix D: Screenshots of UI a number of screenshots with annotations displaying the application's UI, such as Screens for picture input and greeting Display of the forecast result in real time (defect kind and grade) Language choices and user preferences feedback submission form that is embedded within the application The proposed tomato quality assessment system's creation, assessment, and usability testing are supported by these appendices. Appendix A: Examples of Dataset Pictures Sample tomato photos with quality grades (A, B, C, Rejected) and flaws (cracks, rot, shriveling) noted. Code Snippets in Appendix B Data preprocessing, model training, and API integration are all done with Python code. Appendix C: Template for User Feedback Form Both qualitative and quantitative user opinions are gathered through a structured survey. UI Screenshots in Appendix D Interface of a mobile application displaying the features for user interaction, prediction result display, and image input screen.

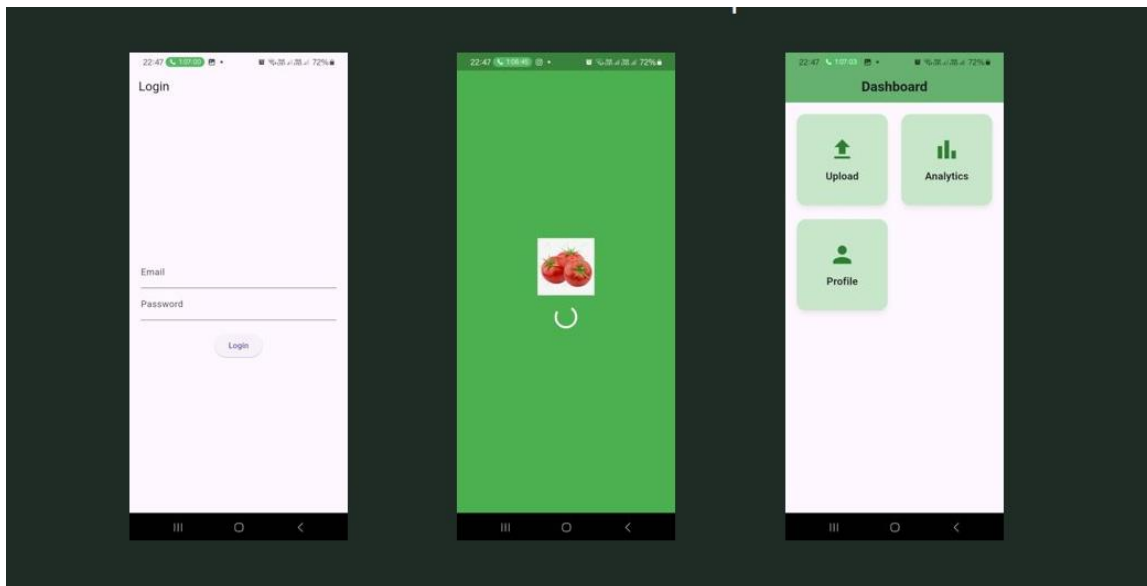


FIGURE 17 UI SCREENSHOTS

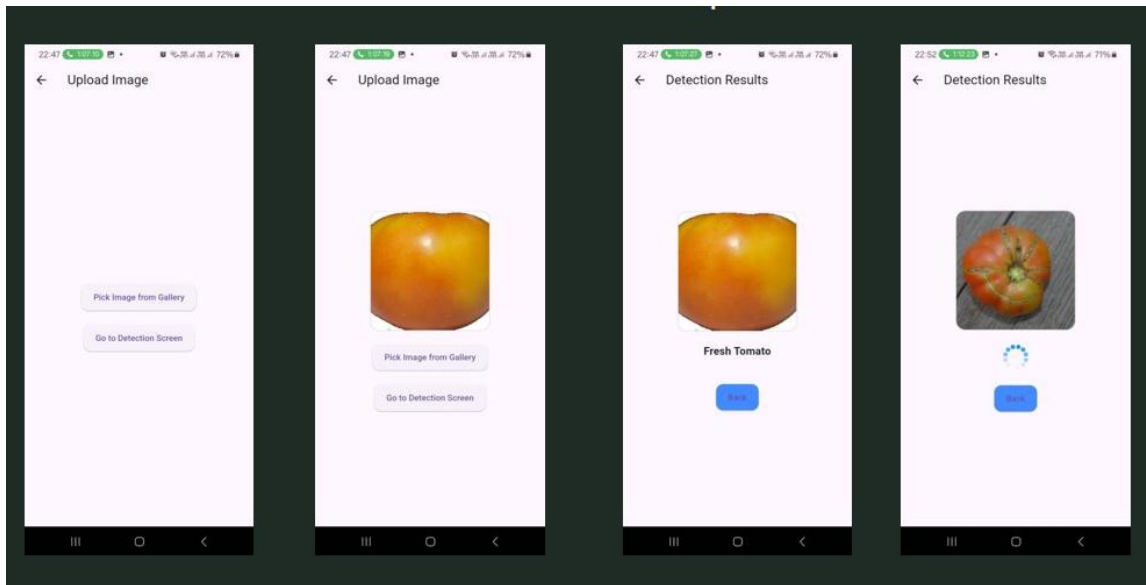


FIGURE 18 UI SCREENSHOTS

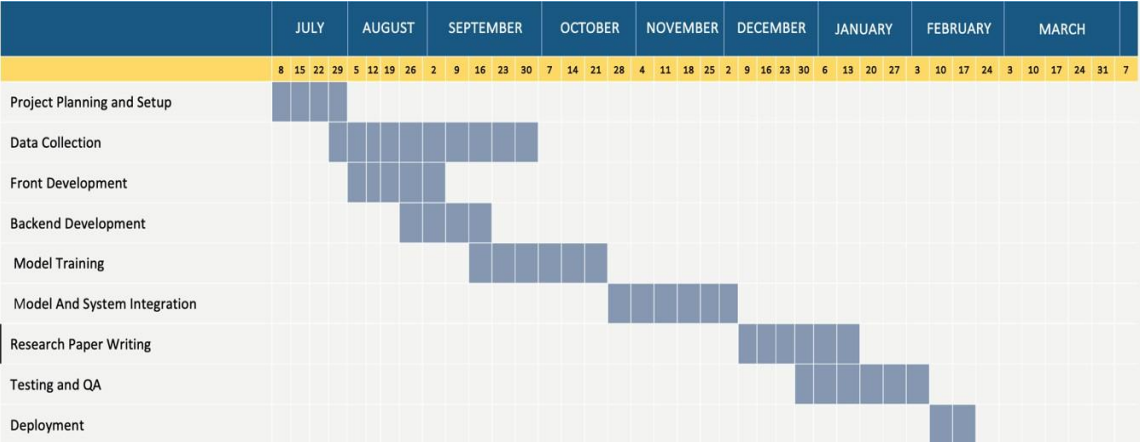


FIGURE 19 GANTT CHART

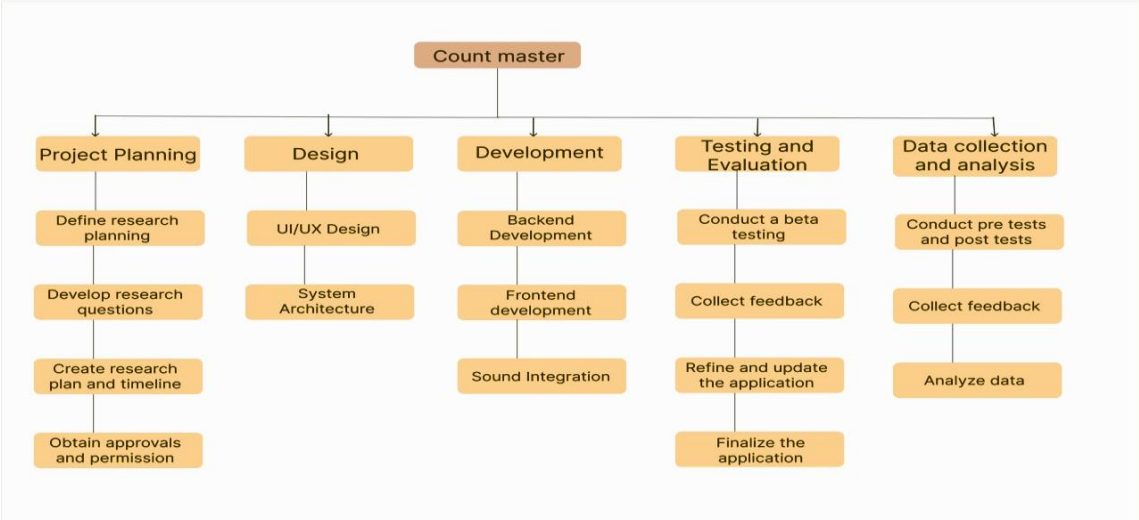


FIGURE 20 WORK BREAKDOWN STRUCTURE

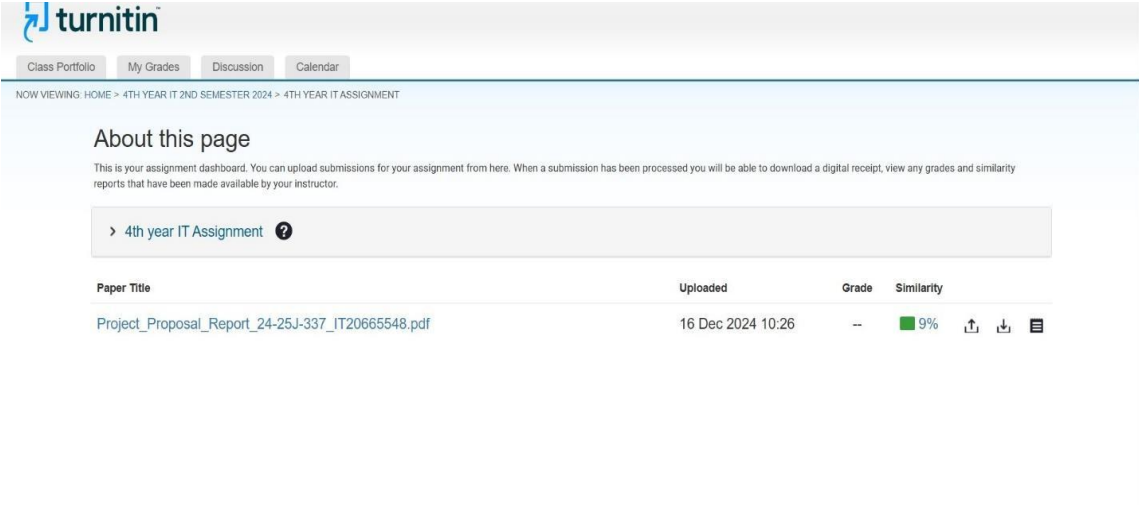


FIGURE 21 PLAGIARISM REPORT