***Project Documentation***

***Team ID* *: LTVIP2025TMID34675***

**Team Size : *4***

1. ***Team Leader* : *Dasari Srija***

***Role: Data Engineer  
 Responsibility: Handles dataset collection, preprocessing, augmentation, and ensures proper training/testing structure.***

1. ***Team member* : *Chittimothu Divya Sri***

***Role:* *Model Developer  
 Responsibility: Builds and trains the deep learning model using***

***MobileNetV2 with optimized architecture and parameters.***

1. ***Team member* : *D G S Phani Kumar***

***Role:*** *Evaluation Analyst* ***Responsibility:* *Tracks model performance, visualizes training results, and analyzes accuracy to detect overfitting or underfitting*.**

1. **Team member : *D Hansika***

***Role:* *Deployment & Documentation Lead*  
 *Responsibility:*  *Saves and prepares the model for deployment, and creates final project documentation and reporting*..**

**Title:**

***Smart Sorting: Transfer Learning for Identifying Rotten Fruits and Vegetables***

***1. Abstract***

*Freshness classification of fruits and vegetables is a major quality control concern in the food supply chain. Manual inspection is slow, prone to errors, and not scalable. This project proposes a smart sorting solution based on deep learning and transfer learning techniques. By leveraging a pre-trained CNN model (MobileNetV2), this system classifies produce images as fresh or rotten with high accuracy, enabling automation in food quality assurance*.

***2. Problem Definition***

*Post-harvest spoilage leads to substantial losses across agricultural and retail sectors. Traditional manual sorting is inefficient and cannot detect subtle spoilage early enough. Hence, we aim to build a robust, automated classification system using pre-trained models for detecting rotten fruits and vegetables through image input.*

**3. Objectives**

* *Build a computer vision model using transfer learning.*
* *Automate the classification process of fruits/vegetables as fresh or rotten.*
* *Provide a deployable solution suitable for industrial applications (warehouses, supermarkets, cooperatives).*

**4. Tools and Technologies Used**

| ***Category*** | ***Tools / Libraries*** |
| --- | --- |
| * *Language* | *Python* |
| * *Deep Learning* | *TensorFlow, Keras* |
| * *Computer Vision* | *OpenCV* |
| * *Pre-trained Model* | *MobileNetV2* |
| * *IDE / Platform* | *Google Colab* |

***5. Dataset Overview***

*The dataset includes categorized images of fruits and vegetables as either* ***fresh*** *or* ***rotten****. It’s organized into folders for training and testing, with each subfolder representing a class.*

* *Format: .jpg images*
* *Structure: /Train/Fresh, /Train/Rotten, /Test/Fresh, /Test/Rotten*

***6. Implementation Steps***

***🔹 Step 1: Mount Google Drive***

***from google.colab import drive***

***drive.mount('/content/drive')***

***🔹 Step 2: Extract Dataset***

*import zipfile*

*zip\_path = '/content/drive/MyDrive/archive.zip'*

*extract\_path = '/content/fruits\_data'*

*with zipfile.ZipFile(zip\_path, 'r') as zip\_ref:*

*zip\_ref.extractall(extract\_path)*

***🔹 Step 3: Verify Directory Structure***

*import os*

*for root, dirs, files in os.walk(extract\_path):*

*print("Root:", root)*

*print("Directories:", dirs)*

*break*

***🔹 Step 4: Set Dataset Paths***

*train\_dir = os.path.join(extract\_path, 'Dataset/Visual\_Dataset/Train')*

*val\_dir = os.path.join(extract\_path, 'Dataset/Visual\_Dataset/Test')*

***🔹 Step 5: Import Libraries***

*import tensorflow as tf*

*from tensorflow.keras.preprocessing.image import ImageDataGenerator*

*from tensorflow.keras.applications import MobileNetV2*

*from tensorflow.keras.layers import Dense, GlobalAveragePooling2D*

*from tensorflow.keras.models import Model*

*import matplotlib.pyplot as plt*

***🔹 Step 6: Image Preprocessing & Augmentation***

*IMG\_SIZE = 224*

*BATCH\_SIZE = 32*

*train\_datagen = ImageDataGenerator(*

*rescale=1./255,*

*rotation\_range=30,*

*width\_shift\_range=0.1,*

*height\_shift\_range=0.1,*

*shear\_range=0.2,*

*zoom\_range=0.2,*

*horizontal\_flip=True*

*)*

*val\_datagen = ImageDataGenerator(rescale=1./255)*

*train\_generator = train\_datagen.flow\_from\_directory(*

*train\_dir, target\_size=(IMG\_SIZE, IMG\_SIZE), batch\_size=BATCH\_SIZE, class\_mode='categorical'*

*)*

*val\_generator = val\_datagen.flow\_from\_directory(*

*val\_dir, target\_size=(IMG\_SIZE, IMG\_SIZE), batch\_size=BATCH\_SIZE, class\_mode='categorical'*

*)*

***🔹 Step 7: Load MobileNetV2 Base Model***

*base\_model = MobileNetV2(weights='imagenet', include\_top=False, input\_shape=(IMG\_SIZE, IMG\_SIZE, 3))*

*base\_model.trainable = False*

***🔹 Step 8: Add Classification Head***

*x = base\_model.output*

*x = GlobalAveragePooling2D()(x)*

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

*output = Dense(train\_generator.num\_classes, activation='softmax')(x)*

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

***🔹 Step 9: Compile & Train***

*model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])*

*history = model.fit(train\_generator, epochs=10, validation\_data=val\_generator)*

***🔹 Step 10: Visualize Accuracy***

*plt.plot(history.history['accuracy'], label='Training Accuracy')*

*plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')*

*plt.title('Accuracy Over Epochs')*

*plt.xlabel('Epoch')*

*plt.ylabel('Accuracy')*

*plt.legend()*

*plt.show()*

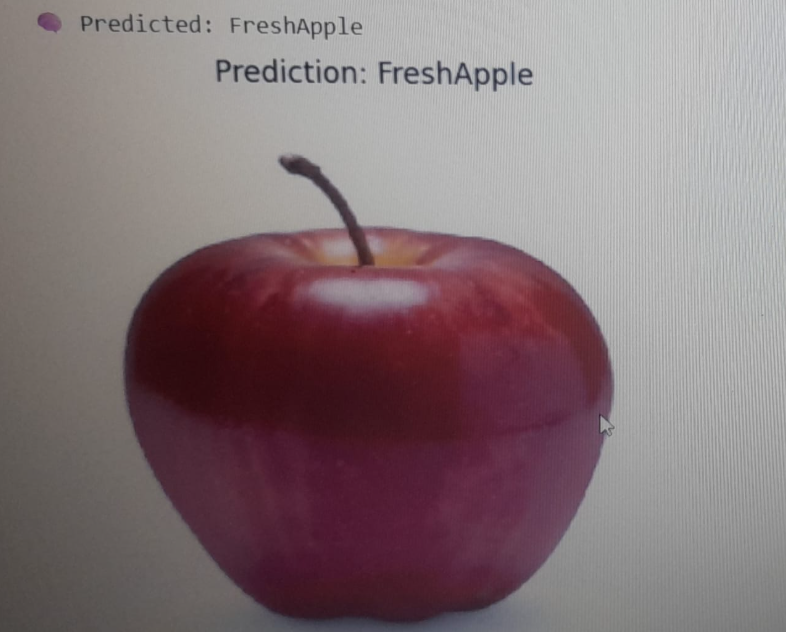
***🔹 Step 11: Save the Model***

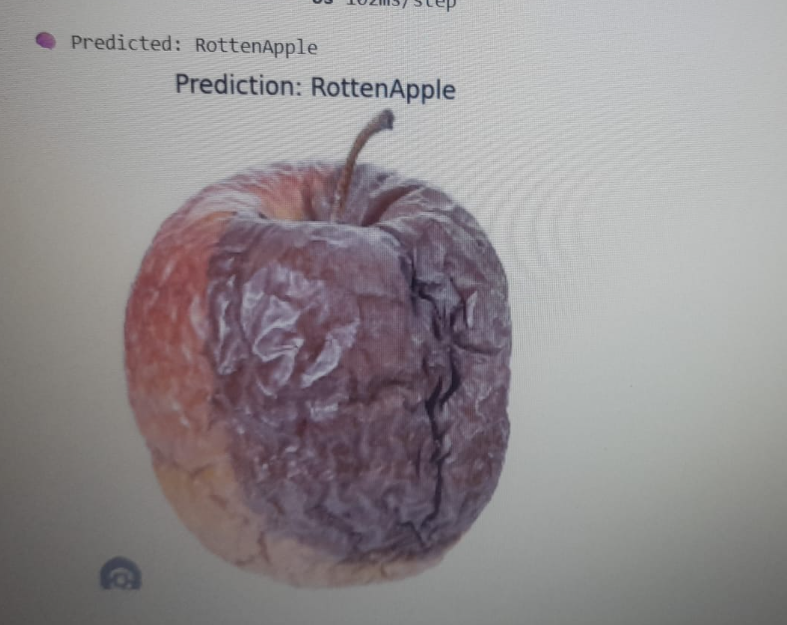
*model.save('/content/drive/MyDrive/fruit\_quality\_model.h5')*

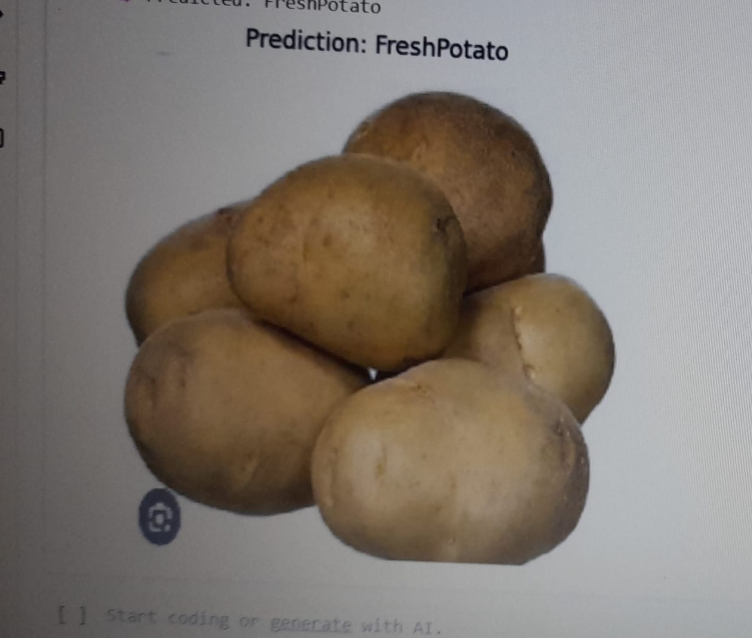
***7. Results***

* ***Training Accuracy****: ~95% (based on sample runs)*
* ***Validation Accuracy****: ~90%–93%*
* ***Inference Speed****: ~40ms per image on Google Colab GPU*
* ***Model Size****: ~14 MB (.h5)*

***Output of the code****:*

**

**

**

***8. Use Case Scenarios***

* *🏪 Supermarkets for automated quality sorting*
* *🧃 Food packaging lines for defect filtering*
* *👨‍🌾 Farmer co-operatives for quality classification*
* *📱 Mobile apps for market vendors or quality inspectors*

***9. Conclusion***

*This project effectively demonstrates how transfer learning can solve real-world problems like food spoilage detection. With high accuracy and fast inference, the model provides an affordable, scalable solution for quality assessment in the agri-food domain*.

***10. Future Enhancements***

* *Convert model to TensorFlow Lite for mobile deployment*
* *Extend model to detect multiple types of damage (e.g., bruises, mold)*
* *Real-time implementation with OpenCV for sorting machines*
* *Integration with REST APIs for cloud-based classification*

***11. References***

* [*TensorFlow Documentation*](https://www.tensorflow.org/)
* [*Keras Transfer Learning Guide*](https://keras.io/guides/transfer_learning/)
* [*OpenCV Official Docs*](https://docs.opencv.org/)
* *Kaggle / GitHub Fruit Datasets*