**SMART SORTING: TRANSFER LEARNING FOR IDENTIFYING ROTTEN FRUITS AND VEGETABLES**

**1. Introduction :**

vegetable Overview of the problem of fruit and spoilage. Importance of automatic sorting. Brief mention of transfer learning and its relevance to image classification in agriculture.

Team I D :

LTVIP2025TMID35904

TEAM MEMBERS :

A.Sumanth

Bodireddy Revanth

T.Venkata Reddy

P.V.Narendra

**2. Project Overview :**

Goals and objectives of the system. Description of how the model identifies rotten produce using machine learning. Real-world applications and benefits (e.g., food quality control, waste reduction).

**3. Architecture :**

Explanation of system components:

* Data collection and preprocessing
* Transfer learning model (e.g., ResNet, EfficientNet)
* Backend services (API)
* Frontend interface
* Deployment strategy

**4. Setup Instructions :**

Steps to install dependencies and configure the environment:

* Hardware and software requirements
* Python version
* Virtual environment setup
* Installation of packages (via requirements.txt

**5. Folder Structure :**

Explain directory layout. Example:

bash

CopyEdit

/smart-sorting/

│

├── data/ # Raw and processed datasets

├── models/ # Saved model weights

├── src/ # Source code (training, preprocessing, inference)

├── api/ # API logic (FastAPI, Flask, etc.)

├── ui/ # Frontend files

├── tests/ # Unit and integration tests

├── requirements.txt # Python dependencies

└── README.md

**6. Running the Application :**

Command-line steps or scripts to:

* Train the model
* Start the API
* Run the UI
* Example input/output

**7. API Documentation :**

Description of endpoints, using Swagger or manual documentation. Example:

* POST /predict: Uploads an image and returns the prediction
* GET /health: Returns service status

**8. Authentication :**

(Optional) Describe how secure access is managed, e.g., via JWT tokens or API keys, if implemented.

**9. User Interface :**

Overview and screenshots of the UI. Key features:

* Uploading images
* Viewing results
* User-friendly indicators (rotten/fresh status)

**10. Testing :**

Coverage of:

* Unit tests for core logic
* Integration tests for model + API
* Tools used (e.g., pytest, unittest)
* Test datasets

**11. Known Issues :**

List current limitations or bugs. Examples:

* Misclassification under poor lighting
* Limited to trained fruit/vegetable classes

**12. Future Enhancements :**

Suggestions for improvement:

* Real-time video analysis
* Larger, more diverse dataset
* Support for multi-class classification (e.g., degrees of ripeness)
* Edge device deployment (Raspberry Pi, Jetson Nano)

**13.Demo Link :** [**https://drive.google.com/file/d/1twJATBtbTxx\_kZld16PzJV6y-SG3Vx9C/view?usp=drivesdk**](https://drive.google.com/file/d/1twJATBtbTxx_kZld16PzJV6y-SG3Vx9C/view?usp=drivesdk)

**14.Ppt Link :**

**hVQlr9kOboZUhttps://drive.google.com/file/d/1FpmuskKZWezx9ZM9oBCI/view?usp=drivesdK**

15.Git Hub Link :

**https://github.com/brevanth31/smart-sorting-transfer-learning-for-identifying-rotten-fruits-and-vegetables**