Ideation Phase
Technologies and Frameworks

Date	19 February 2026
Team ID	LTVIP2026TMIDS77295
Project Name	Smart Sorting: Transfer Learning for Identifying Rotten Fruits and Vegetables

Technologies and Frameworks

1. Programming Language: Python

- The core of the project (model training, prediction, backend logic) is implemented in Python.
- Python is chosen because it has rich support for machine learning (TensorFlow/Keras) and web frameworks (Flask), along with many utilities for data handling and scientific computing.

---

2. Web Framework: Flask

- Flask is a lightweight Python web framework used to build the backend of the Smart Sorting web application.
- Responsibilities:
    - Defines routes such as /, /about, /predict, and /contact.
    - Handles HTTP requests (GET/POST), file uploads, and communication with the machine-learning model.
    - Uses Jinja2 templating to render dynamic HTML pages (Home, About, Predict, Result, Contact).
- Flask is well-suited for this project because it is simple, flexible, and integrates easily with TensorFlow models.

---

3. Deep Learning Framework: TensorFlow and Keras

- TensorFlow with its high-level API Keras is used for building, training, and saving the deep learning model.
- Key uses:
    - Loading the MobileNetV2 pre-trained model (transfer learning) to classify fruits/vegetables as fresh or rotten.
    - Defining custom layers on top of MobileNetV2 for binary classification.
    - Training the model using model.fit() with image generators and saving weights to healthy_vs_rotten.h5.

- TensorFlow/Keras is ideal because it provides ready-made pre-trained models, GPU support, and a clean API for experimentation.

---

4. Pre-trained Model: MobileNetV2

- MobileNetV2 is a convolutional neural network architecture pre-trained on ImageNet, used here via Keras applications as the base model for transfer learning.
- **Advantages:**
    - Lightweight and efficient, making predictions fast even on CPU.
    - Already trained on millions of images, so it can recognize generic features like edges, colors, and shapes that are useful for fruit images.
- **In this project:**
    - MobileNetV2 is loaded without the top classification layer.
    - A custom dense head is added for the Fresh vs Rotten task.
    - Base layers are frozen initially and optionally fine-tuned in later training epochs.

---

5. Data Handling and Scientific Libraries

- **NumPy**
    - Used for numerical operations, array manipulation, and preparing tensors to feed into the model.
- **Pillow (PIL) or OpenCV (if used)**
    - Used for opening, resizing, and converting images before prediction.
    - Ensures images match the input format expected by MobileNetV2 (e.g., 224×224 RGB).
- **scikit-learn**
    - Provides evaluation utilities such as confusion matrix and classification report to analyze model performance.
    - Helps in performance testing by generating metrics like precision, recall, and F1-score for Fresh and Rotten classes.
- **Matplotlib and Seaborn**
    - Used to visualize training/validation accuracy and loss curves.
    - Used to plot confusion matrices and other performance graphs for the report.

---

6. Frontend Technologies: HTML, CSS, JavaScript

- **HTML5**
    - Structures the web pages (Home, About, Predict, Result, Contact).
    - Forms are used on the Predict page to upload images and send them to the Flask backend.
- **CSS3 (and optionally Bootstrap)**
    - Provides styling, layout, colors, font choices, and responsive design.
    - Makes the UI user-friendly, with cards, buttons, navigation bar, and proper alignment.
    - If Bootstrap is used, it simplifies responsive design with grid system and pre-built components.
- **JavaScript**
    - Enhances interactivity, for example:
        - Previewing the selected image before uploading.
        - Handling client-side validations (e.g., alert if no file is selected).
        - Implementing dark/light theme toggles or small UI effects.
    - Runs in the browser and works together with Flask-rendered HTML pages.

---

7. Project and Version Control Tools

- **Git (if used)**
    - Tracks changes in code, notebooks, and configuration files.
    - Supports collaboration within the team and maintains history of model versions and bug fixes.
- **Virtual Environment (venv)**
    - Isolates project dependencies (Flask, TensorFlow, etc.) from global Python packages.
    - Helps keep the environment reproducible across different systems.

---

8. File and Template Structure (Flask Specific)

- **Templates Folder**
    - Contains HTML files rendered by Flask (index.html, about.html, predict.html, result.html, contact.html).

    - Uses Jinja2 placeholders to inject dynamic data such as prediction label and confidence score.
- **Static Folder**
    - Contains static resources like CSS files, JavaScript files, and images.
    - Flask serves these assets to style pages and handle client-side behavior.