Project Design Phase-II

Smart Sorting : Transfer Learning For Identifying

Rotten Fruits and Vegetables

(Technology Stack (Architecture & Stack))

Date	19 FEB 2026
Team ID	LTVIP2026TMIDS65633
Project Name	Smart sorting: transfer Learning For Identifying Rotten Fruits and Vegetables
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Architecture Description:

A web-based intelligent classification system that allows users to upload images of fruits/vegetables, which are processed through a pretrained transfer learning model (e.g., MobileNet, VGG16) to classify them as **fresh** or **rotten**. Results are displayed via a UI, with admin controls for dataset management and retraining.

Architecture Includes:

- User interacts via web UI
- Frontend sends image to backend via Flask API
- Backend processes image using CNN transfer learning model
- Classification result is returned and displayed
- Admin manages datasets and retrains models
- All data (images, logs, feedback) is stored locally/cloud

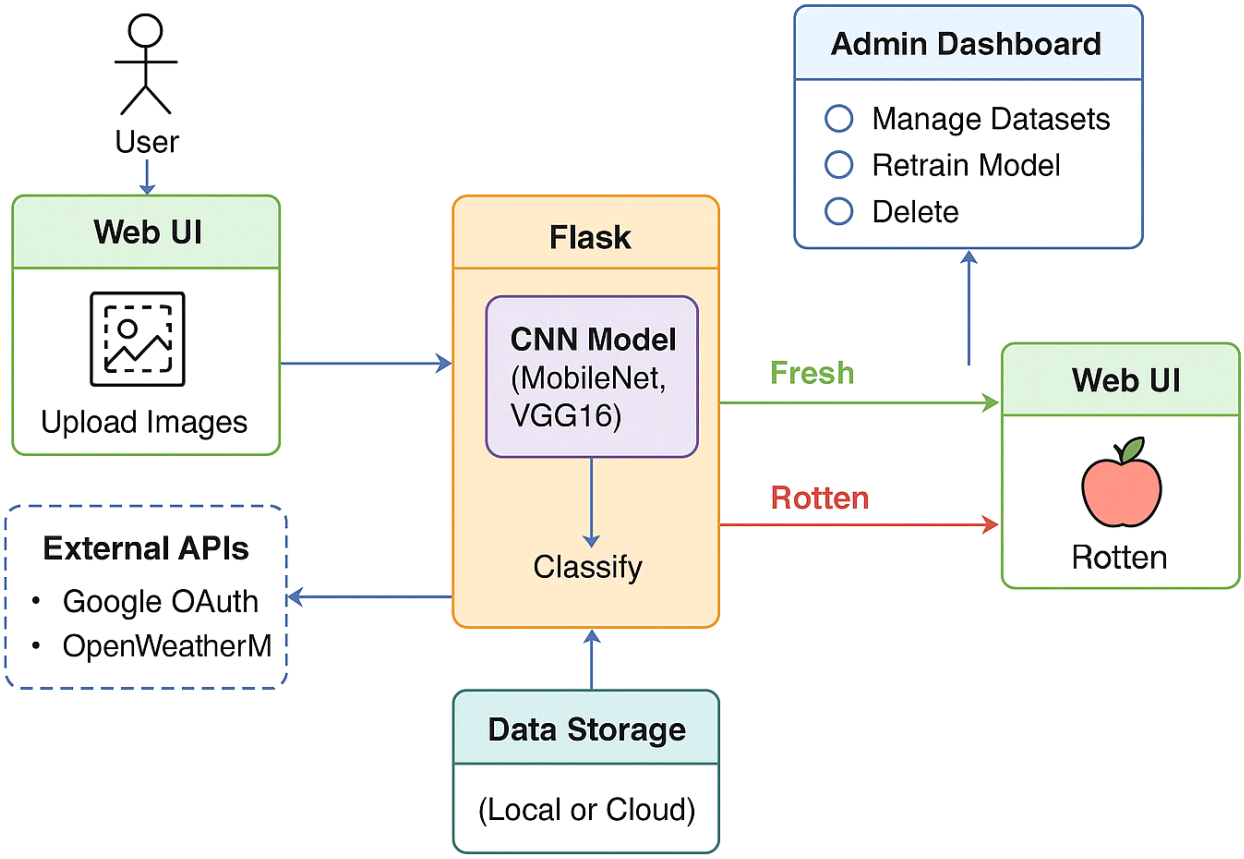

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
•	User Interface	Interface for image upload, result viewing, and admin panel	HTML, CSS, JavaScript, Bootstrap
•	Application Logic-1	Image upload and preprocessing logic	Python + Flask
•	Application Logic-2	Image classification using ML model	Transfer Learning (MobileNet / VGG16)
•	Application Logic-3	Dataset feedback and retraining module	Python, TensorFlow / Keras
•	Database	Storage of prediction results and feedback	SQLite / MySQL
•	File Storage	Storage for image uploads and training data	Local Filesystem / AWS S3
•	Machine Learning Model	Transfer learning for fruit/vegetable classification	CNN (MobileNet / ResNet / VGG16, etc.)
•	Infrastructure	Model runs on local server; scalable to cloud deployment	Localhost / Flask Server / Google Colab / AWS EC2.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
•	Open-Source Frameworks	Deep learning, backend, frontend frameworks	TensorFlow, Flask, Bootstrap, React (opt.)
•	Security Implementations	Authentication, data access control	OAuth 2.0, HTTPS, SHA-256
•	Scalable Architecture	Can shift from single-tier to 3-tier or cloud-microservice-based system	Flask APIs, AWS EC2, Docker (optional)
•	Availability	Can host via cloud servers with failover	AWS / GCP / Heroku
•	Performance	Optimized image preprocessing, batching, and model caching	TensorFlow Lite (optional), Flask Caching