Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	31 June 2025
Team ID	LTVIP2025TMID35093
Project Name	Smart Sorting & Transfer Learning for Identifying Rotten Fruits and Vegetables
Maximum Marks	4 Marks

Technical Architecture:

Technical Architecture Diagram

The architecture is designed using a **3-tier structure** with the following flow:

```
csharp
CopyEdit
[User Interface (Mobile/Web)]
        ↓
[Application Layer (Python-based backend)]
        ↓
[ML Model (TensorFlow / Transfer Learning)]
        ↓
[Database & File Storage]
        ↓
[Mechanical Control Layer (Servo / Arduino)]
```

- Data is captured via camera or file upload from the UI.
- Processed using the ML model hosted in the application logic layer.
- Based on prediction, actuator (servo/motor) is triggered for sorting.
- Results are logged in a database and visible on the dashboard.

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Web & Mobile UI for users/admins	HTML, CSS, JavaScript, Bootstrap, React JS
2	Application Logic-1	Backend logic for classification & dashboard	Python (Flask or Django)
3	Application Logic-2	Motor/Servo control for sorting action	Arduino/C++ with Python Serial Communication
4	Application Logic-3	Data logging, user management	Python, SQLite/MySQL
5	Database	Local database for logging results and user data	SQLite / MySQL
6	Cloud Database	Scalable database for logs and analytics	Firebase / IBM Cloudant
7	File Storage	Store images temporarily for classification	Local Filesystem / Firebase Storage
8	External API-1	Notify admin of errors/issues	Twilio / Email SMTP API
9	External API-2	Optional: Get market trends for produce	RapidAPI (e.g., Fruit Price API)
10	Machine Learning Model	Transfer Learning for Rotten vs Fresh classification	TensorFlow / Keras with MobileNetV2 or ResNet
11	Infrastructure	Local for MVP, scalable to cloud or edge devices	Raspberry Pi / IBM Cloud / AWS EC2

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology

1	Open-Source Frameworks	All backend, frontend, ML libraries are open-source	Flask, React, TensorFlow, Arduino IDE
2	Security Implementations	Password hashing, admin login control, OTP/email confirmation, access limits	SHA-256, OAuth2, Firebase Auth, SSL
3	Scalable Architecture	3-tier structure; microservice-ready logic for classification and control	Flask APIs, Docker containers
4	Availability	Local + Cloud option, fault-tolerant motor control, backups	Load balancing (Cloud), Redundant logging
5	Performance	Optimized ML inference, limited image size, cache recent results	TensorFlow Lite, Redis (optional), CDN