Project Report Format

1. **INTRODUCTION**

1.1 Project Overview

2. Smart Sorting is an AI-based web application designed to classify images of **fruits and vegetables** into two categories: **healthy** and **rotten**. It leverages a Convolutional Neural Network (CNN) model built using **TensorFlow/Keras** (specifically using **VGG16** as the base) to perform accurate image classification.

3. The purpose is to assist in waste management, food quality control, and sustainability by automating the sorting process. The application allows users to upload an image and get real-time predictions about the freshness of produce.

1.2 Purpose

The purpose of the **Smart Sorting** project is to develop an intelligent system that automates the classification of fruits and vegetables into **healthy** and **rotten** categories using image recognition. This solution aims to reduce **manual errors**, **labor costs**, and **food waste** by providing a fast, scalable, and accessible method for assessing produce quality — benefiting supply chains, households, and waste management systems.

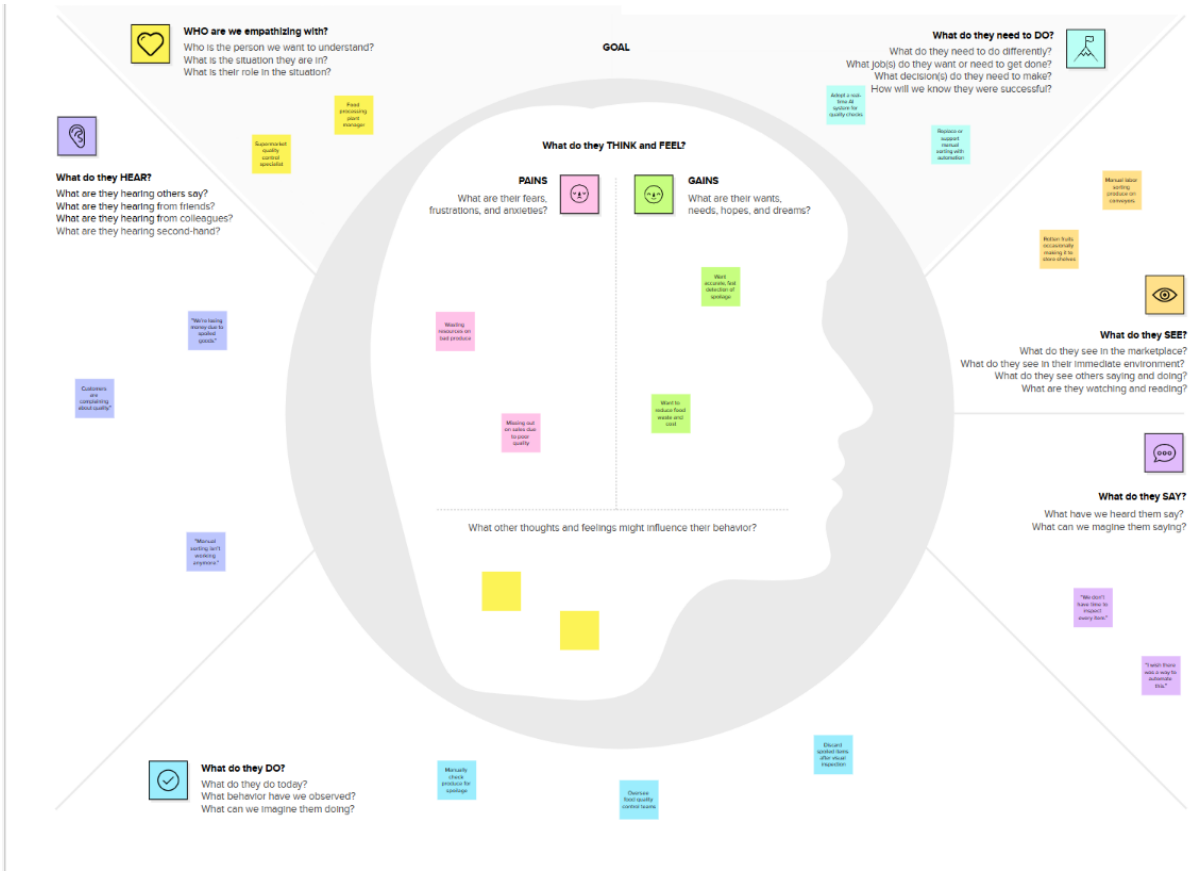
2. **IDEATION PHASE**

2.1 Problem Statement

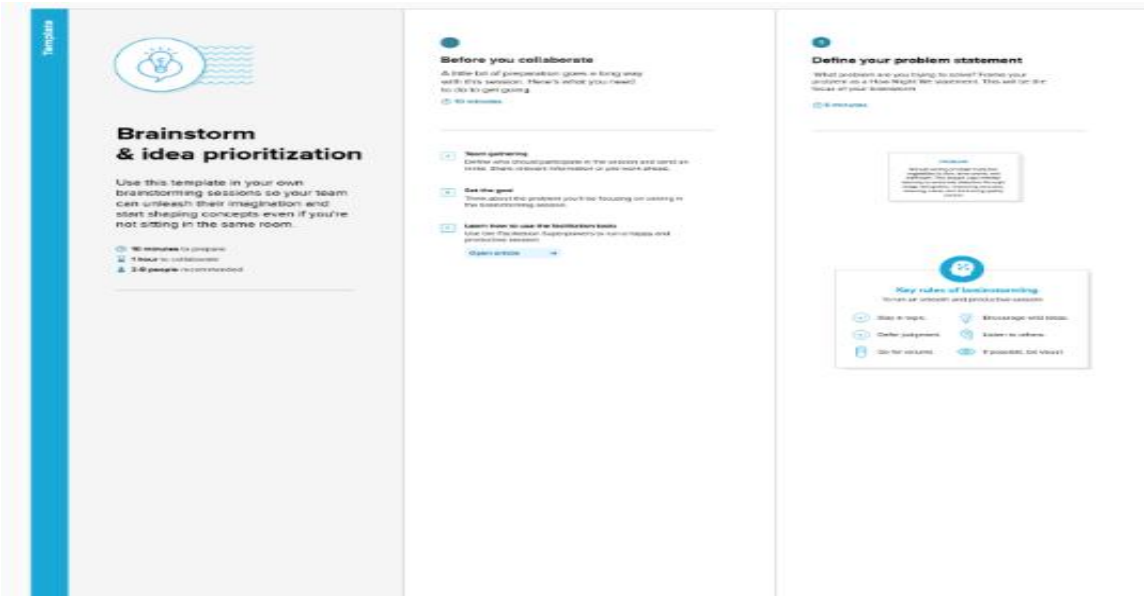
Manual sorting of fruits and vegetables is labor-intensive, time-consuming, and prone to human error. There is a lack of scalable solutions that can assist in quick and reliable identification of spoiled produce, which leads to food waste and inefficiencies in the supply chain.

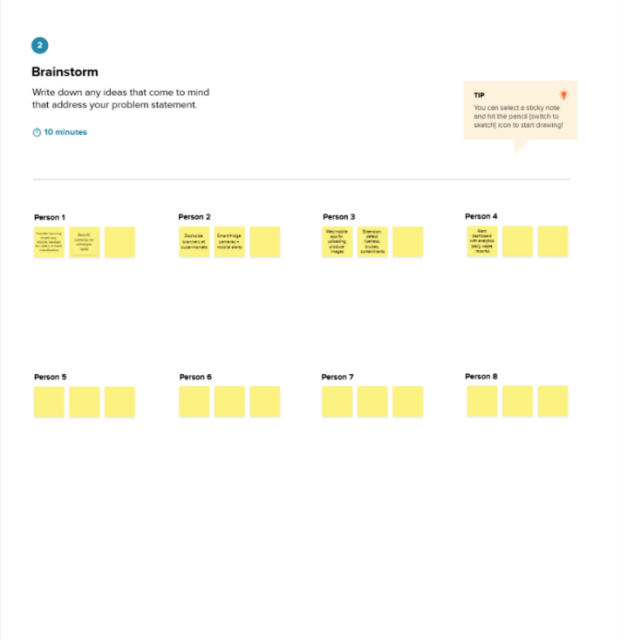
A food processing plant manager	Quickly detect and remove rotten produce	Manual sorting is slow and inefficient	There's no real-time automated detection system	Frustrated by repeated waste
A food safety officer	Ensure only fresh fruits/vegetables reach customers	Rotten produce is sometimes missed	Workers rely only on visual checks	Worried about customer complaints

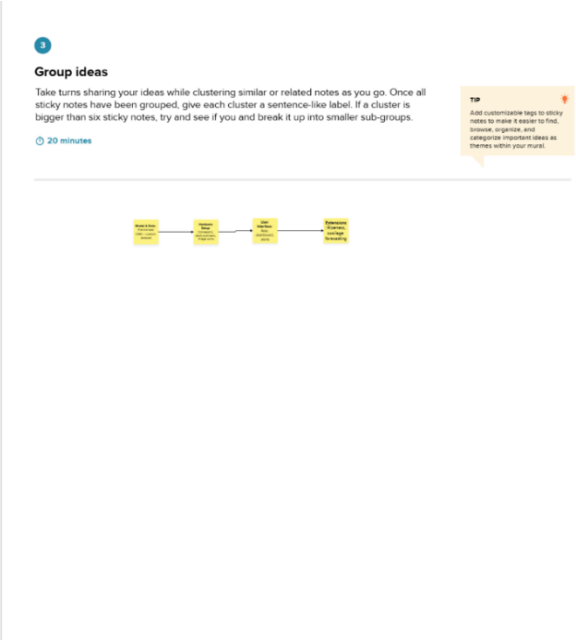
2.1 Empathy Map Canvas

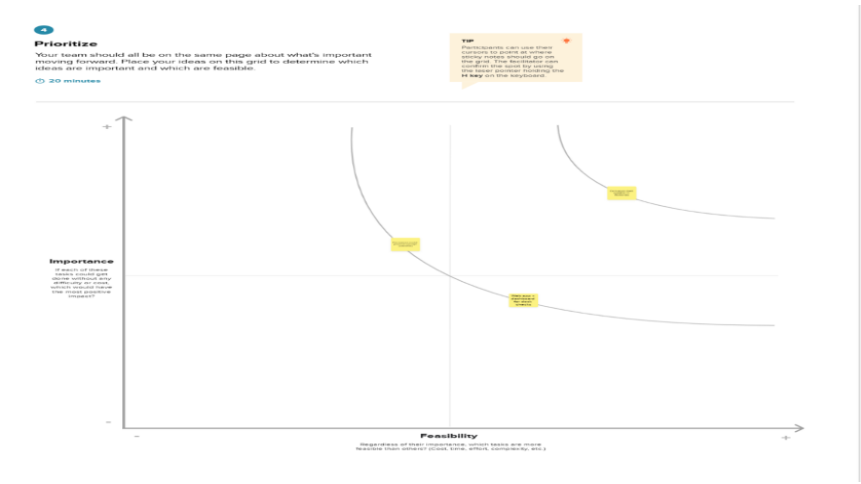


2.2 Brainstorming









3 REQUIREMENT ANALYSIS

3.1 Customer Journey map

Customer Journey Map: Smart Sorting for Rotten Fruits and Vegetables

Steps	Experiences	Interactions	Touchpoints	Places	People	Goals & Motivations	Positive Moments	Negative Moments	Areas of Opportunity
Entice	Users discover Smart Sorting via home appliance ads, blogs, or grocery apps.	Click on product link or QR code on fridge.	Website, smart fridge UI, app ads.	Home, online retail sites.	Shoppers, tech bloggers.	Help me reduce waste at home.	Excited by tech that saves food.	Skepticism about reliability.	Use videos/testimonials to build trust.
Enter	User explores product details and how it fits into the kitchen.	Scrolls product page, watches demo.	Product videos, feature tabs.	Online shop, appliance store.	Sales agents, family members.	Help me understand if it's easy to use.	Clear visuals and use-cases.	Jargon or tech-heavy terms.	Offer simple explanations, demos.
Engage	Installs and begins using smart sorting in fridge.	Connects app, configures alerts.	Mobile app, fridge camera UI.	Kitchen, smartphone.	Family, app support.	Help me detect spoilage in time.	Feels empowered and informed.	Setup can be confusing.	Provide guided setup walkthrough.
Exit	System identifies rotten items and sends alert.	Receives notification on mobile.	App alerts, fridge LED.	Home (kitchen).	User only.	Help me act before food goes bad.	Saves money and reduces waste.	Too many false positives.	Allow user to confirm before discarding.
Extend	User shares insights or tips with friends, rates the app.	Shares app link, writes review.	Social media, in-app feedback.	Home, online.	Friends, app community.	Help others benefit like I did.	Feels like a sustainability hero.	May forget to leave feedback.	Gamify reviews or offer small rewards.

3.2 Solution Requirement

Functional Requirements:

Following are the functional requirements of the proposed solution.

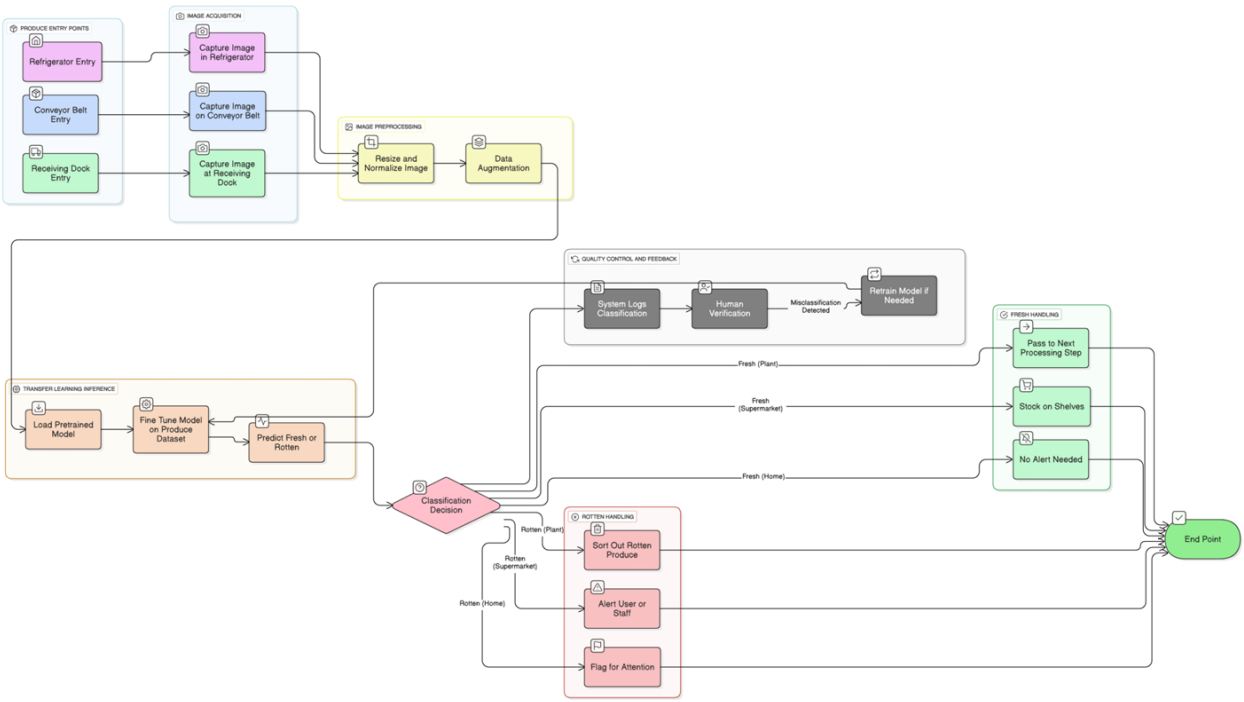
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Image Upload & Prediction	Upload image via web form Show predicted class (healthy/rotten)
FR-2	Model Integration	Load pre-trained VGG16 model (.h5) Predict using trained model
FR-3	User Interface	Provide clean, responsive interface Display image preview after upload
FR-4	Data Visualization	Show sample images from each class Display basic stats (class count)

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Simple browser interface usable by anyone without coding knowledge
NFR-2	**Security**	Secure image uploads handled with file type and size checks
NFR-3	**Reliability**	System produces consistent results for same input images.
NFR-4	**Performance**	Image prediction within 2–3 seconds on standard laptop/PC.
NFR-5	**Availability**	Can be run offline or locally; no cloud dependency.
NFR-6	**Scalability**	Can be trained with additional classes or deployed on mobile/IoT.

3.3 Data Flow Diagram



User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Web User (Customer)	Upload Image for Prediction	USN-1	As a user, I can upload an image of a fruit or vegetable through the web browser	Image is successfully uploaded and displayed on screen	High	Sprint-1
		USN-2	As a user, I receive a prediction of the image as “healthy” or “rotten”	Prediction label appears instantly below the image	High	Sprint-1
		USN-3	As a user, I want the system to prevent invalid file types	If invalid file is uploaded, system shows error	Medium	Sprint-2
	View Result	USN-4	As a user, I want to see a clean summary of the result (predicted class)	Clear label and confidence score is shown	Medium	Sprint-1
	User Interface	USN-5	As a user, I want the app to be easy to use and mobile-friendly	App looks neat, buttons work, responsive design	High	Sprint-1
	Upload from Different Devices	USN-6	As a user, I want to upload from mobile/laptop without compatibility issues	App works on mobile, tablet, and laptop browsers	Low	
Admin	Model Integration	USN-7	As an admin, I want to load the pre-trained model into the Flask server	Model is successfully loaded at app startup	High	
Admin	Dataset Visualization	USN-8	As an admin, I want to visualize class distribution and image samples	Output of matplotlib shows sample images and class count	Medium	
Admin	Monitor Predictions	USN-9	As an admin, I want to test system with various inputs for validation	Outputs are consistent for known test inputs	Medium	
Admin	Deploy App	USN-10	As an admin, I want to deploy app locally via Anaconda/Flask	App runs at localhost and predicts as expected	High	

3.4 Technology Stack

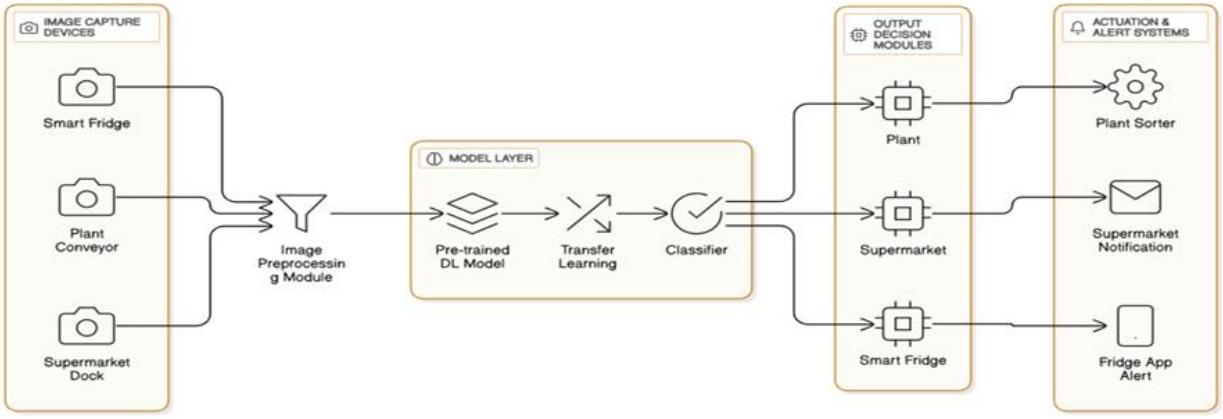


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web UI for uploading images and viewing predictions	HTML, CSS, JavaScript, Bootstrap
2.	Application Logic-1	Flask-based server logic to handle upload, model prediction	Python, Flask
3.	Application Logic-2	Image preprocessing and resizing before feeding to model	OpenCV, TensorFlow/Keras
4.	Application Logic-3	Not used in this project	-
5.	Database	No traditional DB required; optional logging via flat files	JSON / CSV (optional), SQLite (if expanded)
6.	Cloud Database	Not applicable (runs locally or on personal system)	-
7.	File Storage	Uploaded images and trained .h5 model file	Local Filesystem
8.	External API-1	Not used in this project	-
9.	External API-2	Not used in this project	-
10.	Machine Learning Model	To classify fruits/vegetables as healthy or rotten using image data	VGG16 (Transfer Learning) – Keras / TensorFlow
11.	Infrastructure (Server / Cloud)	Application runs on user system using Anaconda or can be deployed to cloud	Local (Flask), Cloud-compatible (Heroku, Render)

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Uses open libraries and frameworks	Flask (Python), Keras, TensorFlow, Bootstrap
2.	Security Implementations	Restricts file type upload, basic validation, no external sharing of files	File type validation, Flask secure upload methods
3.	Scalable Architecture	Can be scaled by deploying on cloud with retraining options	Flask Microservice; Cloud container ready
4.	Availability	Local app, can be made highly available via cloud and CDN	Render / Heroku / AWS or Azure Hosting (optional)
5.	Performance	Fast image prediction with pre-loaded model; small load handled easily	TensorFlow optimized, low latency, local model use

4 PROJECT DESIGN

4.1 Problem Solution Fit

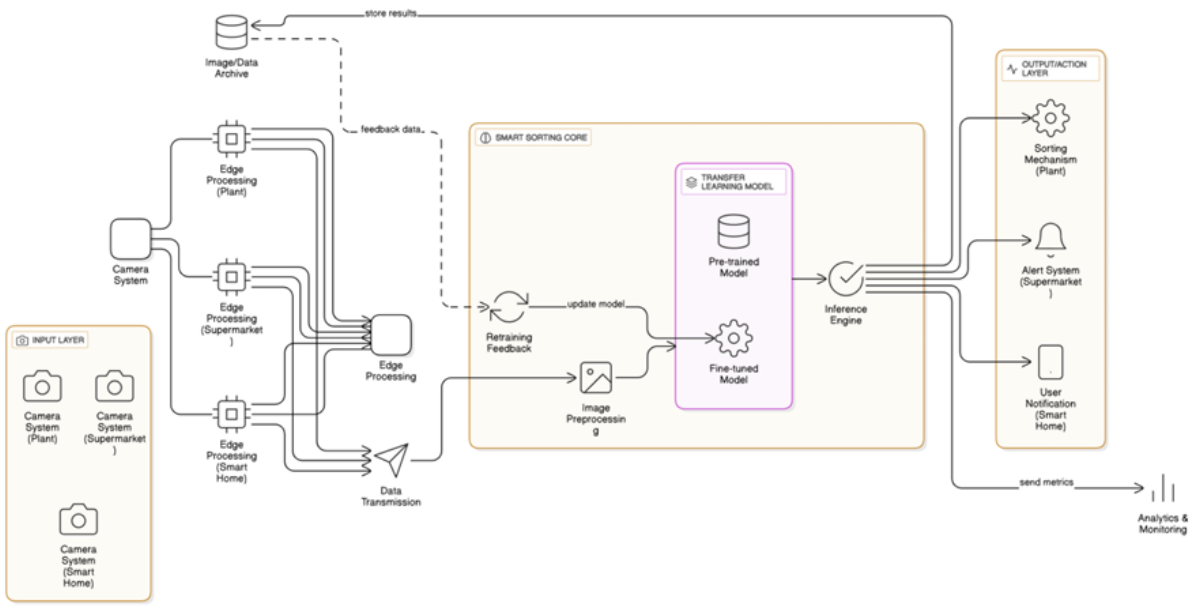
4.2 Proposed Solution

Proposed Solution Template:

Project team shall fill the following information in the proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	A large percentage of fruits and vegetables are wasted due to poor sorting and delayed detection of spoilage. Manual inspection is time-consuming, inconsistent, and not scalable for farmers, vendors, or warehouses. There is a need for a fast, automated, and accurate solution to classify produce as healthy or rotten using technology.
2.	Idea / Solution description	SmartSorting is an AI-powered image classification system that uses a deep learning model (VGG16) to detect and classify fruits and vegetables as **healthy or rotten**. Users can upload images through a web interface, and the system instantly provides a prediction. It is trained on 29 classes using a dataset of 29,000+ images and can be deployed via a lightweight Flask application.
3.	Novelty / Uniqueness	Unlike traditional sorting mechanisms, SmartSorting is low-cost, uses **pre-trained transfer learning (VGG16)**, and can be operated through a simple web browser. It can handle multiple fruit and vegetable types, making it flexible and suitable for various environments (farms, markets, warehouses). The solution also supports real-time predictions and can be adapted to mobile or IoT devices.
4.	Social Impact / Customer Satisfaction	The system reduces food waste, increases supply chain efficiency, and ensures better quality control. Farmers, small vendors, and cold storage units can benefit without needing expensive hardware. The simplicity and accuracy of the tool ensure ease of use and confidence in quality assurance.
5.	Business Model (Revenue Model)	The solution can be monetized as a **subscription-based SaaS platform** for agricultural companies, marketplaces, and warehouse chains. A freemium model can also be offered for small vendors. Additional services like bulk analysis, mobile integration,

4.3 Solution Architecture



5 PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Dataset Setup & Preprocessing	USN-1	As a team, we will download, unzip, and organize the dataset into train/test folders.	2	High	
Sprint-1	Visualization	USN-2	As a user, I want to visualize the classes and sample images to understand the data.	2	High	
Sprint-1	Model Design	USN-3	As a user, I want to use transfer learning (VGG16) to build an image classification model.	3	High	
Sprint-1	Data Augmentation	USN-4	As a user, I want to apply data augmentation to improve model generalization.	2	Medium	
Sprint-2	Training the Model	USN-5	As a user, I want to train the model and save it as an .h5 file.	3	High	
Sprint-2	Flask App	USN-6	As a user, I want to upload an image and get predictions using a web interface.	3	High	
Sprint-3	UI/UX Enhancement	USN-7	As a user, I want the app interface to be clean, user-friendly, and responsive.	2	Medium	

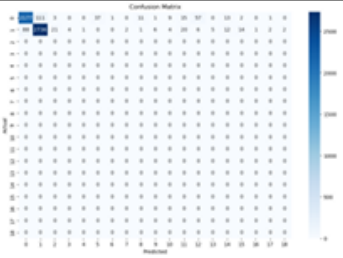
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Testing Predictions	USN-8	As a user, I will test with real-world images from the internet or camera.	2	Medium	
Sprint-4	Evaluation Metrics	USN-9	As a user, I want to generate confusion matrix, accuracy, MAE, and MSE.	3	Medium	
Sprint-4	Documentation	USN-10	As a user, I want to document my project and upload it to GitHub.	2	High	

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	4 Days	15 June 2025	18 June 2025	20	18 June 2025
Sprint-2	20	4 Days	19 June 2025	20 June 2025	18	20 June 2025
Sprint-3	20	4 Days	23 June 2025	23 June 2025	19	23 June 2025
Sprint-4	20	4 Days	27 June 2025	30 June 2025	20	30 June 2025

6 FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE -N/A , MSE – N/A, RMSE -N/A , R2 score -N/A **Classification Model:** Confusion Matrix - , Accuray Score- 91.61% & Classification Report – - Precision (avg): ~0.96 - Recall (avg): ~0.92 - F1-score (avg): ~0.94 - Undefined values for classes with 0 samples	
2.	Tune the Model	Hyperparameter Tuning - Not performed Validation Method - Validation Split using ImageDataGenerator (20%)	

7 RESULTS

7.1 Output Screenshots

8 ADVANTAGES & DISADVANTAGES

1. **Automation & Speed**

- Quickly classifies produce without human effort.
- Saves time in sorting and reduces manual inspection errors.

2. **Reduces Food Waste**
    - Early detection of rotting items helps in timely disposal or preservation.
    - Prevents contamination of nearby healthy items.
3. **Affordable & Scalable**
    - Can be deployed using basic hardware (camera + Raspberry Pi or PC).
    - Easily scalable to handle larger datasets and more food categories.
4. **Customizable AI Model**
    - Can retrain with new data for better accuracy over time.
    - Extendable to different fruits, vegetables, or food types.
5. **Educational & Practical**
    - Great demonstration of deep learning, computer vision, and Flask web apps.
    - Useful for farms, warehouses, supermarkets, and homes.

Disadvantages / Limitations

1. **Data Dependency**
    - Accuracy depends on the quality and diversity of the training dataset.
    - May not work well with poor lighting or blurry images.
2. **Binary Limitation**
    - Currently supports only two classes: **healthy** vs **rotten**.
    - Doesn't give detailed information (e.g. type of rot, severity, etc.).
3. **Needs Preprocessing**
    - May require consistent background, lighting, and resolution for optimal predictions.
4. **Not Real-time (yet)**
    - Depending on implementation, may have delay in prediction.
    - Not optimized yet for edge devices or real-time conveyor sorting.
5. **Model Drift Over Time**
    - Over time, changes in fruit appearance (due to seasons or sources) might reduce accuracy unless model is retrained regularly.

9 CONCLUSION

The Smart Sorting project demonstrates the effective use of artificial intelligence and computer vision in addressing real-world problems like food quality assessment. By leveraging a deep

learning model trained to classify healthy and rotten produce, the system offers a fast, efficient, and automated solution to minimize food waste and improve sorting accuracy. Integrated with a user-friendly Flask web application, it showcases how AI can be practically deployed for agricultural and commercial use. While the current model is limited to binary classification, it sets a strong foundation for future enhancements such as multi-class detection, real-time implementation, and integration with IoT devices. Overall, Smart Sorting is a step toward intelligent, sustainable, and scalable food management solutions.

10 FUTURE SCOPE 11. APPENDIX

Source Code(if any)

Dataset Link : https://www.kaggle.com/datasets/muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten

GitHub: https://github.com/Harikhabammidi/SmartSorting-using-transfer-learning.git

Project Demo Link : https://drive.google.com/file/d/1NdrdsyxH5M7SizqwpoTf72wbyBIGvCrk/view?usp=sharing