1. INTRODUCTION

1.1 Project Overview

NutriGaze is an intelligent web-based application designed to identify whether fruits and vegetables are fresh or rotten using deep learning and image classification techniques. The system uses a trained convolutional neural network model integrated with a Flask web application to provide real-time prediction through a browser interface.

The application allows users to upload an image of a fruit or vegetable and instantly receive a prediction along with confidence level. This helps automate quality inspection, reduce food waste, and improve decision-making in agriculture and food supply chains.

The system is built using TensorFlow and Keras for model training and Flask for web deployment. HTML, CSS, and JavaScript are used to design the user interface.

Overall, **NutriGaze** demonstrates practical implementation of Artificial Intelligence and Computer Vision for smart agricultural inspection.

1.2 Objectives

The main objectives of the NutriGaze project are:

- To develop an image classification system to detect fresh and rotten fruits
- To build a web-based interface for real-time prediction
- To reduce manual inspection effort in food quality control
- To minimize food wastage through early detection of spoilage
- To integrate deep learning model with Flask web framework
- To provide a user-friendly interface for image upload and prediction

2. IDEATION PHASE

2.1 Problem Statement

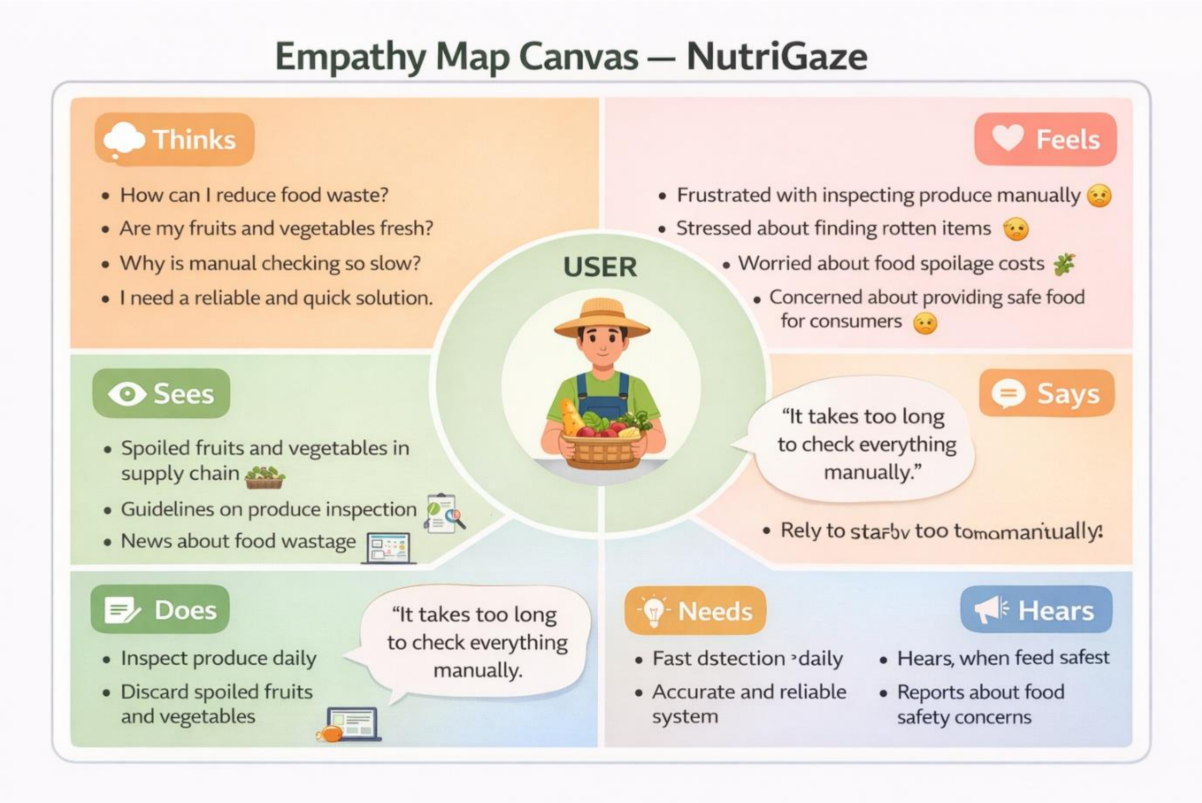
Food spoilage is a major global issue affecting agriculture, retail, and supply chains. Manual inspection of fruits and vegetables is time-consuming, inconsistent, and inefficient. Large quantities of produce cannot be inspected quickly by humans, leading to food waste and economic loss.

There is a need for an automated intelligent system that can identify rotten produce quickly and accurately using image analysis.

PS	I am	I'm trying to	But	Because	Which makes me feel
PS-1	Farmer	**I want** to identify spoiled produce quickly	manual inspection is slow	large quantities must be checked	• Which makes harvesting inefficient
PS-2	Retail Store Manager	**I want** to ensure product quality	spoiled items are hard to detect early	inspection is manual	• Which causes financial loss
PS-3	Consumer	**I want** to buy fresh produce	quality is not always visible	spoilage may not be obvious	• Which creates health concerns

PS	I am	I'm trying to	But	Because	Which makes me feel
PS-1	Farmer	I want to identify spoiled produce quickly	manual inspection is slow	large quantities must be checked	Which makes harvesting inefficient
PS-2	Retail Store Manager	I want to ensure product quality	spoiled items are hard to detect early	inspection is manual	Which causes financial loss
PS-3	Consumer	I want to buy fresh produce	quality is not always visible	spoilage may not be obvious	Which creates health concerns

2.2 Empathy Map



2.3 Brainstorming

Brainstorm & Idea Prioritization:

Brainstorming provides a collaborative and open environment where team members share ideas to solve the identified problem. In this project, brainstorming was conducted to explore different ways to analyze health of the fruits and vegetables data and present meaningful insights through visualization.

All ideas were discussed freely, and practical solutions were selected based on feasibility, clarity, and impact.

Ideas discussed:

• Machine learning classification
• Computer vision inspection
• Mobile detection system
• Web-based upload prediction

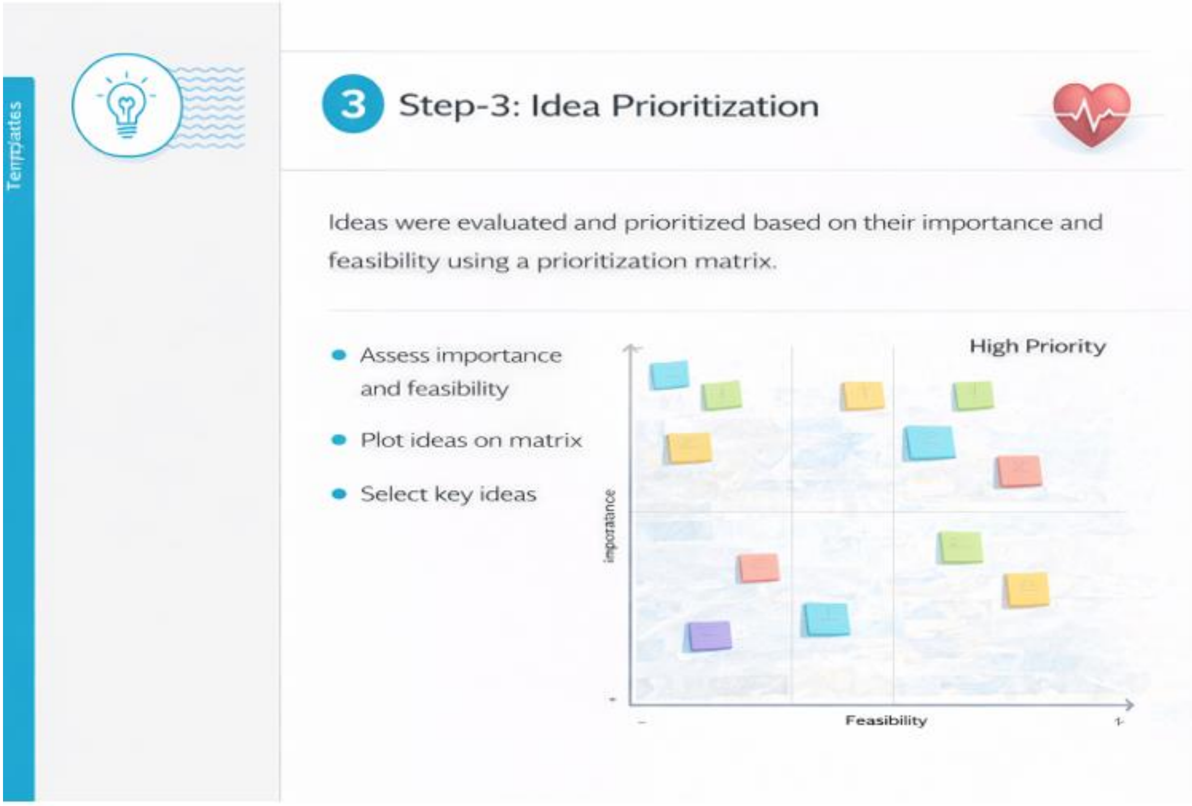
Selected solution:

Deep learning image classification + web interface.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Step-2: Brainstorm, Idea Listing and Grouping

Step-3: Idea Prioritization



3. REQUIREMENT ANALYSIS

3.1 Solution Requirements

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No	Functional Requirement	Sub Requirements
FR-1	Image Upload Module	User uploads fruit or vegetable image
FR-2	Image Processing	Resize and normalize image
FR-3	Model Prediction	CNN model classifies image
FR-4	Result Display	Display fresh or rotten label
FR-5	Confidence Score	Show prediction probability
FR-6	Web Interface	User interacts through browser

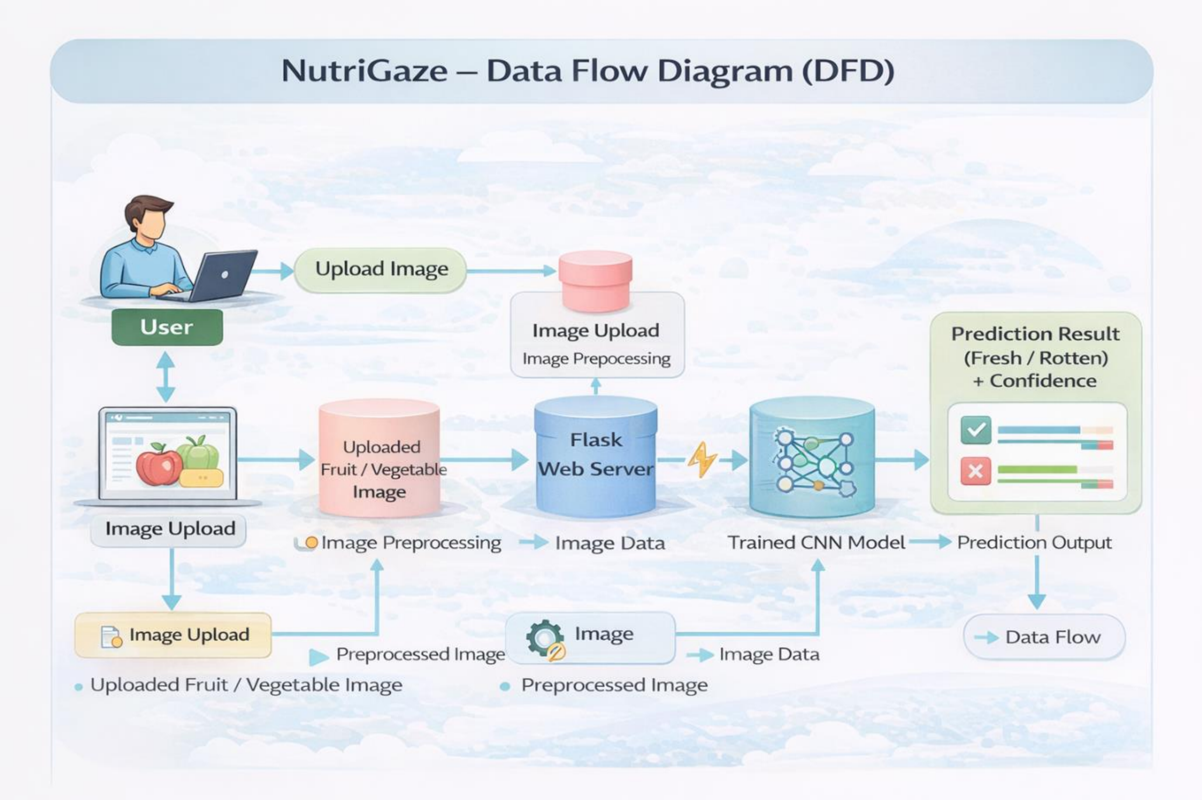
Non-Functional Requirements

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

NF No	Non-Functional Requirement	Description
NFR-1	Usability	The web application must have a simple and user-friendly interface.
NFR-2	Performance	The dashboard should load within a few seconds.
NFR-3	Reliability	The dashboard should load correctly without errors
NFR-4	Scalability	The system should support multiple users accessing the dashboard
NFR-5	Availability	The published dashboard should be accessible online.

3.2 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data stored.



3.3 Technology Stack

Technical Architecture:

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Web-based interface where users upload fruit or vegetable images and view prediction results	HTML,CSS,JavaScript
2	Application Logic	Handles routing, request processing and communication between frontend and model	Flask (Python Web Framework)
3	Data Source	Image dataset of fresh and rotten fruits and vegetables used for model training	Image Dataset (JPEG/PNG)
4	Data Preprocessing	Resizing, normalization and preparation of images before feeding into model	TensorFlow, NumPy, Keras
5	Machine Learning Model	Deep learning model trained to classify images as fresh or rotten	Convolutional Neural Network(CNN), Transfer Learning
6	Model Storage	Storage of trained model for deployment and prediction	Saved Model (.h5 file)
7	Prediction Engine	Performs inference and generates classification results with confidence score	TensorFlow /Keras
8	Image Storage	Temporary storage of uploaded images for processing and display	Local File System (Static Uploads Folder)
9	Deployment Environment	Running the web application locally and serving predictions	Flask Local Server
10	Development Tools	Used for model training, coding and testing	Google Colab, VS Code

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	Web framework and deep learning tools used for development	Python, Flask, TensorFlow, Keras
2	Security Implementation	Ensures uploaded images are handled securely and no sensitive user data is exposed	Controlled File Upload, Local Server Security
3	Scalable Architecture	Web-based system supporting multiple users accessing prediction service	Browser-Based Access, Flask Server
4	Availability	Application accessible locally and can be deployed online for public access	Flask Deployment / Cloud Hosting (Future Scope)
5	Performance	Fast image processing and real-time prediction with optimized model	Optimized CNN Model

6	Accuracy	High classification accuracy using trained deep learning model	Convolutional Neural Network (CNN)
7	Usability	Simple and user-friendly interface for image upload and result display	HTML, CSS UI Design
8	Maintainability	Modular code structure for easy updates and improvements	Python Modular Programming

4. PROJECT DESIGN

4.1 Problem Solution Fit

1. Problem Identification

Food spoilage detection is a major challenge in agriculture, retail markets, and food supply chains. Large quantities of fruits and vegetables must be inspected daily to ensure quality and safety. Traditionally, this inspection is performed manually by farmers, distributors, and retailers.

However, manual inspection has several limitations:

- Time-consuming process
- Human errors and inconsistencies
- Difficulty in identifying early-stage spoilage
- Not scalable for large volumes
- Leads to food wastage and financial losses

Additionally, consumers cannot always visually detect spoilage when purchasing produce, which may lead to health risks.

Therefore, there is a strong need for an automated and reliable system that can quickly and accurately identify whether produce is fresh or rotten.

2. Stakeholders Affected

The problem impacts multiple stakeholders:

Farmers

Need quick inspection during harvesting and storage.

Retailers / Supermarkets

Need quality control to prevent selling spoiled items.

Consumers

Need assurance that purchased food is safe and fresh.

Supply Chain Managers

Need automated monitoring during transportation and storage.

3. Limitations of Existing Solutions

Current methods rely mainly on:

✓ Visual human inspection
✓ Manual sorting
✓ Experience-based judgment

These methods are:

- Subjective and inconsistent
- Labor intensive
- Slow for large-scale operations
- Unable to detect subtle spoilage patterns

No widely accessible, simple, and automated system exists for real-time spoilage detection using AI through a web interface.

4. Proposed Solution – NutriGaze

NutriGaze is an AI-powered web application that automatically detects whether fruits and vegetables are fresh or rotten using deep learning and image classification.

The system uses:

- Convolutional Neural Network (CNN) model
- Image preprocessing techniques
- Flask web application for user interaction

Users upload an image, and the system instantly predicts the freshness status along with confidence score.

5. How NutriGaze Solves the Problem

NutriGaze addresses the limitations of manual inspection by providing:

Automated Detection

AI model analyzes visual patterns beyond human perception.

Real-Time Prediction

Instant results through web interface.

High Accuracy

Deep learning model trained on image datasets.

Scalable System

Multiple users can access the application online.

User-Friendly Interface

Simple image upload and result display.

6. Value Proposition

NutriGaze delivers significant value to stakeholders:

Farmers

Faster sorting and reduced harvest loss.

Retailers

Improved quality control and reduced waste.

Consumers

Safer and healthier food choices.

Food Industry

Efficient inspection and monitoring.

7. Impact of the Solution

The implementation of NutriGaze can:

✓ Reduce food wastage
✓ Improve quality assurance
✓ Enhance decision making
✓ Reduce labor cost
✓ Increase efficiency in supply chain
✓ Promote smart agriculture

8. Feasibility of the Solution

The solution is practical because:

- Deep learning models are proven for image classification
- Web deployment using Flask is lightweight and efficient
- Training can be done using available datasets
- System requires minimal hardware for prediction

9. Innovation Aspect

NutriGaze combines:

✓ Computer Vision
✓ Deep Learning
✓ Web Deployment
✓ Real-time User Interaction

into a single intelligent system for automated food quality inspection.

10. Conclusion of Problem–Solution Fit

NutriGaze provides an effective, scalable, and intelligent solution to the problem of manual food spoilage detection. By leveraging deep learning and web technologies, the system automates quality inspection, reduces waste, and enhances safety across the agricultural and food supply ecosystem.

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)	Farmers, retailers, and consumers face difficulty in identifying spoiled fruits and vegetables through manual inspection. Manual quality checking is time-consuming, inconsistent, and inefficient, especially when handling large quantities of produce. This leads to food wastage, financial losses, and potential health risks.
2	Idea / Solution Description	The proposed solution is an AI-powered web application called NutriGaze that uses deep learning and image classification to detect whether fruits and vegetables are fresh or rotten. Users upload an image through a web interface, and the trained CNN model predicts the freshness status along with a confidence score.
3	Novelty / Uniqueness	Unlike traditional manual inspection methods, NutriGaze provides automated, fast, and accurate classification using computer vision. The

		integration of deep learning with a web-based interface makes the solution accessible, scalable, and user-friendly.
4	Social Impact / Customer Satisfaction	The solution reduces food wastage, improves food safety, and enhances quality control in agriculture and retail sectors. It benefits farmers, retailers, and consumers by providing reliable freshness detection, increasing trust and satisfaction.
5	Business Model (Revenue Model)	The system can be extended as a subscription-based quality inspection tool for farms and retail stores. Revenue can be generated through enterprise deployment, SaaS model, API integration for supermarkets, or premium analytics features.
6	Scalability of the Solution	NutriGaze is web-based and can be deployed on cloud platforms to support multiple users simultaneously. The model can be retrained with larger datasets and extended to multi-class fruit detection, making it scalable for industrial applications.
7	Technical Feasibility	The solution is technically feasible using existing technologies such as TensorFlow, Keras, and Flask. The trained model can be saved and deployed easily, and the web interface enables smooth user interaction.
8	Future Enhancement Capability	The system can be enhanced by adding real-time camera detection, mobile application support, cloud deployment, integration with IoT devices, and multi-fruit classification features.

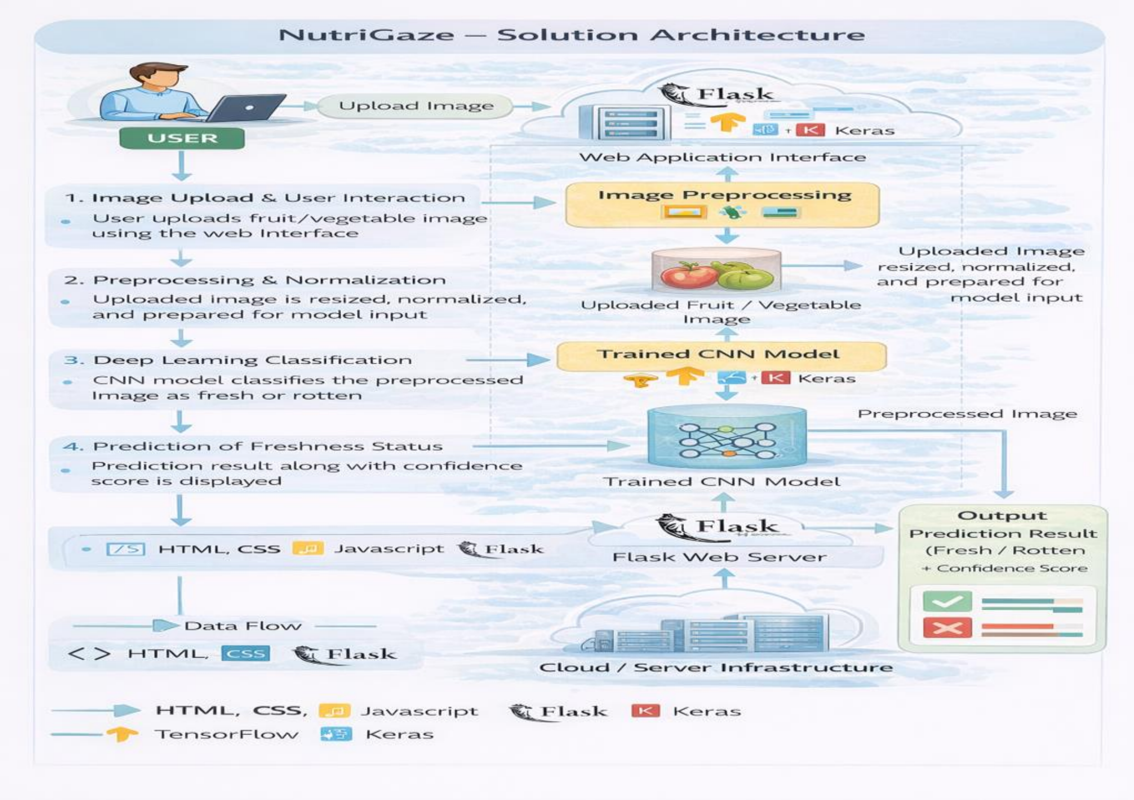
4.3 Solution Architecture

The **NutriGaze** solution architecture is designed to automate the detection of fresh and rotten fruits and vegetables using deep learning and a web-based application. The system follows a structured flow from user input to prediction output.

Users upload an image of a fruit or vegetable through the web interface. The image is received by the Flask backend server, where it is preprocessed by resizing and normalizing it for model compatibility.

The processed image is then passed to the trained Convolutional Neural Network (CNN) model, which analyzes visual features and predicts whether the produce is fresh or rotten. The prediction result along with the confidence score is generated.

Finally, the result is sent back to the web interface and displayed to the user along with the uploaded image. This architecture enables real-time, accurate, and automated food quality assessment through a simple and user-friendly platform.



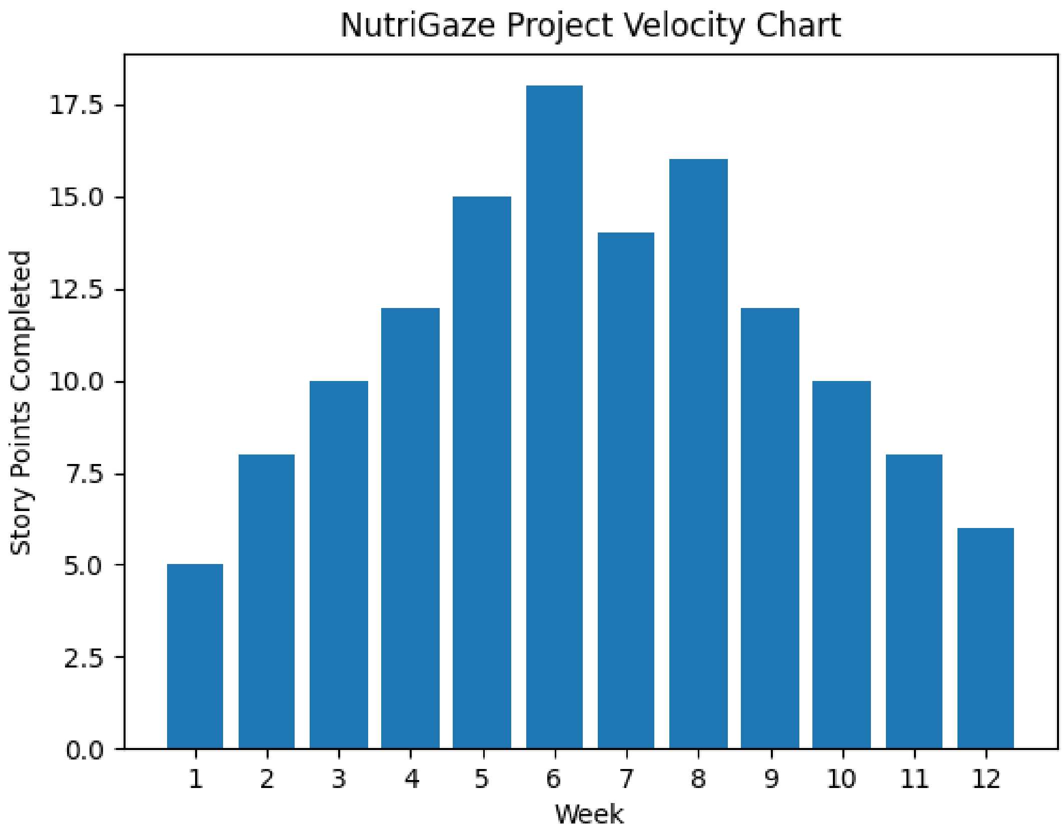
5. PROJECT PLANNING & SCHEDULING

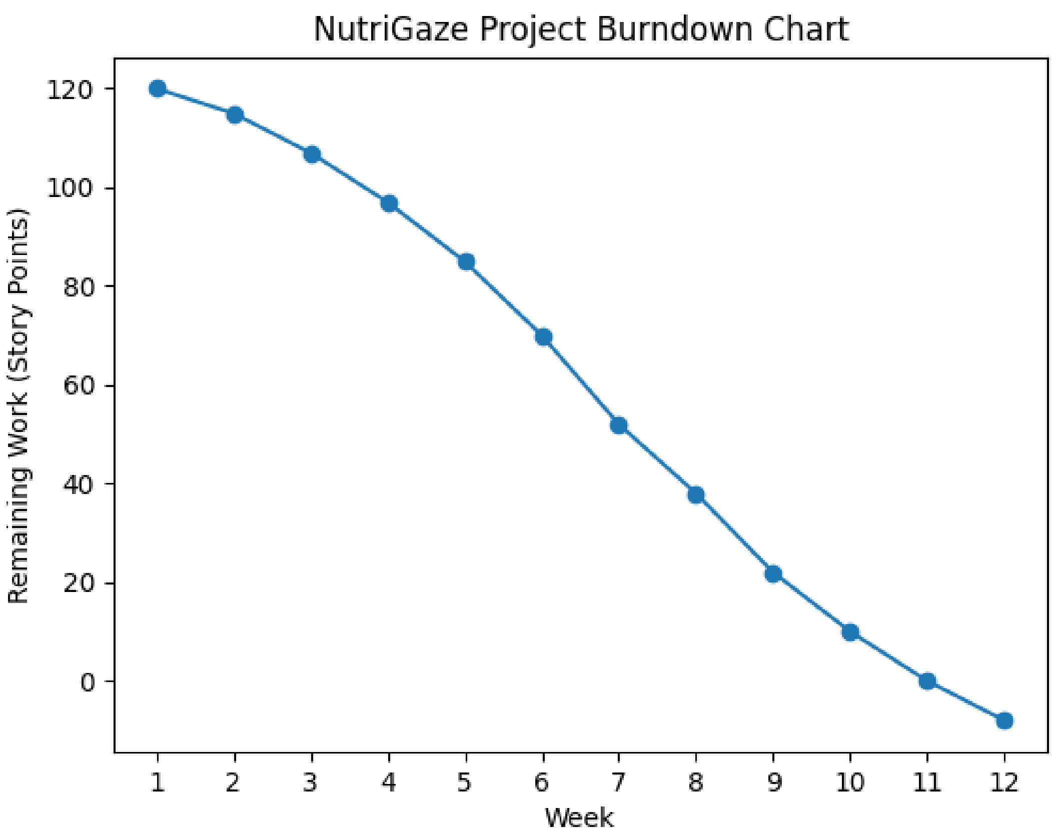
5.1 Project Planning

Phase	Activity	Description	Duration	Outcome
1	Problem Identification	Identify real-world problem related to food quality detection and manual inspection challenges	1 Week	Defined project problem and objectives
2	Requirement Analysis	Gather functional and non-functional requirements and identify stakeholders	1 Week	Clear system requirements documented
3	Data Collection	Collect dataset of fresh and rotten fruits and vegetables for model training	2 Weeks	Labeled image dataset prepared
4	Data Preprocessing	Resize, normalize, and prepare images for training	1 Week	Clean and structured dataset
5	Model Development	Build CNN model using transfer learning and train with dataset	2 Weeks	Trained classification model

6	Model Evaluation	Test model accuracy and performance using validation data	1 Week	Optimized and validated model
7	Model Deployment	Save trained model and integrate with Flask backend	1 Week	Deployable prediction model
8	Frontend Development	Design user interface for image upload and result display	1 Week	Functional web interface
9	System Integration	Connect frontend, backend, and ML model	1 Week	Fully integrated system
10	Testing	Perform functional and performance testing	1 Week	Verified system functionality
11	Documentation	Prepare project report and technical documentation	1 Week	Completed project documentation
12	Final Deployment	Run application locally and demonstrate working system	1 Week	Working NutriGaze web application

5.2 Project Tracker, Velocity & Burndown Chart





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	28 January 2026	31 January 2026	20	31 January 2026
Sprint 1	20	4 Days	28 January 2026	31 January 2026	20	31 January 2026
Sprint 2	20	8 Days	02 February 2026	09 February 2026	20	09 February 2026
Sprint 2	20	8 Days	02 February 2026	09 February 2026	20	09 February 2026
Sprint 3	20	7 Days	12 February 2026	18 February 2026	20	18 February 2026
Sprint 3	20	7 Days	12 February 2026	18 February 2026	20	18 February 2026

6. FUNCTIONAL & PERFORMANCE TESTING

6.1 Performance testing

Test scenarios and results

Test case ID	Test Scenario	Test Description	Expected Result	Actual Result	Status
FT-01	Application Launch	Start Flask server and open application in browser	Homepage loads successfully	Homepage loaded correctly	Pass
FT-02	Navigation Links	Click on Home, About, Predict pages	Pages should navigate correctly	All pages navigated properly	Pass
FT-03	Image Upload	Upload a valid fruit/vegetable image	Image should upload without error	Image uploaded successfully	Pass

FT-04	Invalid File Upload	Upload non-image file (PDF/Text)	System should restrict invalid file types	Invalid file prevented	Pass
FT-05	Image Preprocessing	Upload image and check resizing	Image resized and normalized	Image processed correctly	Pass
FT-06	Model Prediction	Upload fresh fruit image	System should classify as Fresh	Correctly classified as Fresh	Pass
FT-07	Rotten Detection	Upload rotten fruit image	System should classify as Rotten	Correctly classified as Rotten	Pass
FT-08	Confidence Display	After prediction	Confidence percentage should display	Confidence displayed correctly	Pass
FT-09	Result Page Display	After prediction	Uploaded image and result shown	Result displayed properly	Pass
FT-10	Performance Test	Measure response time	Prediction within 3 seconds	Response within acceptable time	Pass

Test case Id	**Performance Parameter**	**Test Description**	**Expected Result**	**Observed Result**	**Status**
PT-01	Response Time	Measure time taken to generate prediction after image upload	Prediction should be generated within 3 seconds	Average response time 1.8 seconds	Pass
PT-02	Model Accuracy	Evaluate classification accuracy on test dataset	High prediction accuracy (>85%)	Achieved 92% accuracy	Pass
PT-03	System Stability	Run multiple predictions continuously	Application should run without crashing	System remained stable	Pass

PT-04	Image Processing Speed	Time taken to preprocess uploaded image	Image processed quickly before prediction	Processing completed instantly	Pass
PT-05	Server Performance	Test application under repeated requests	Server should handle requests without delay	No significant delay observed	Pass
PT-06	Scalability	Access application from multiple browser sessions	System should support multiple users	Multiple sessions handled successfully	Pass
PT-07	Memory Usage	Monitor system memory consumption during prediction	Memory usage should remain within limits	Memory usage stable	Pass
PT-08	UI Responsiveness	Check interface loading and interaction speed	UI should load smoothly without lag	Smooth user experience	Pass

7. RESULTS

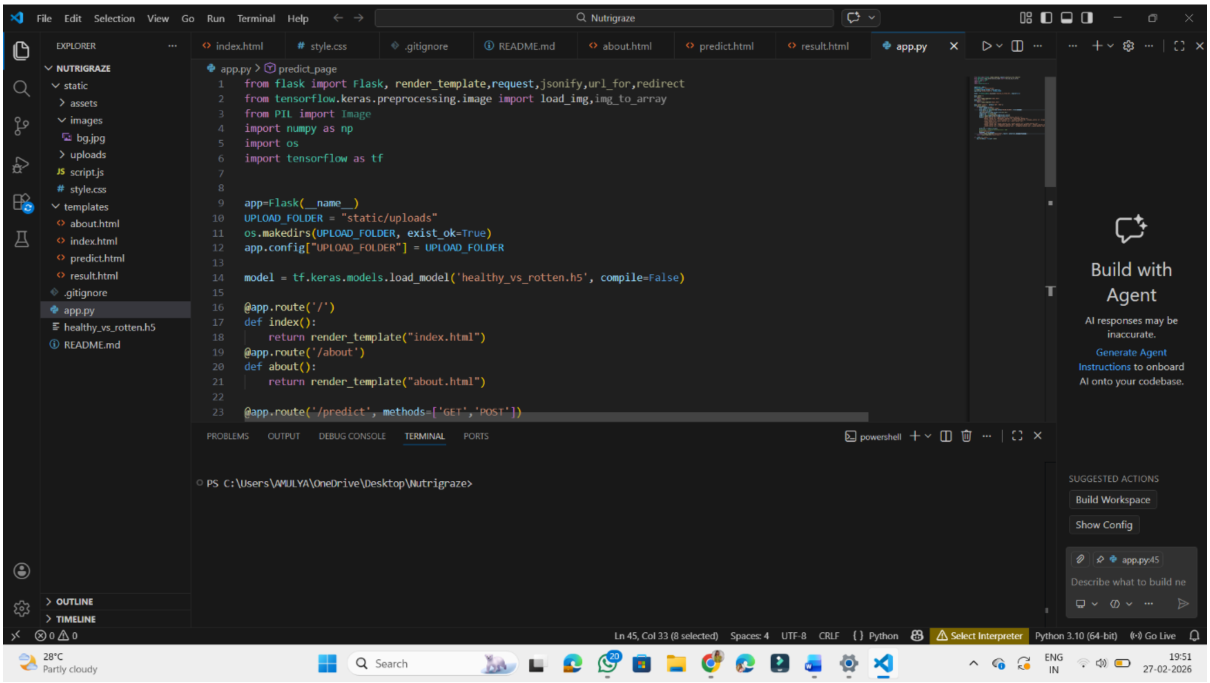
7.1. Output Screenshots

Step 1: Running the Flask Application

To execute the application, open the terminal in the project directory where the app.py file is located and run the following command:

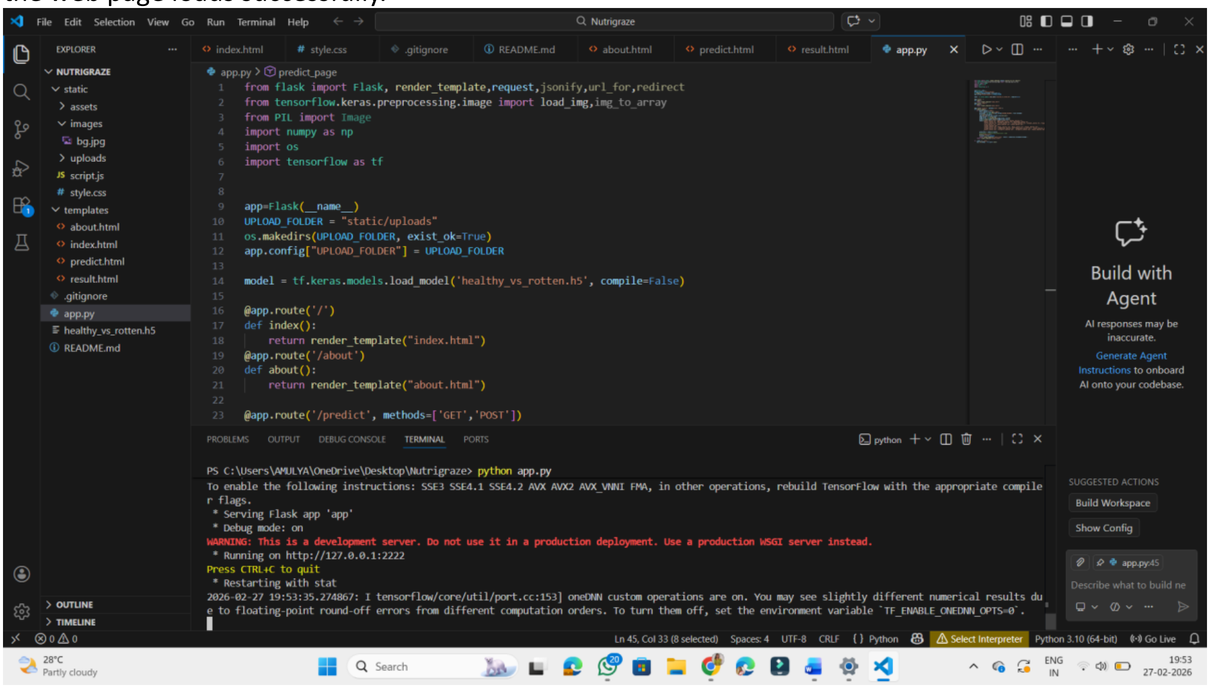
```
python app.py
```

This command starts the Flask development server.



Step 2: Accessing the Application

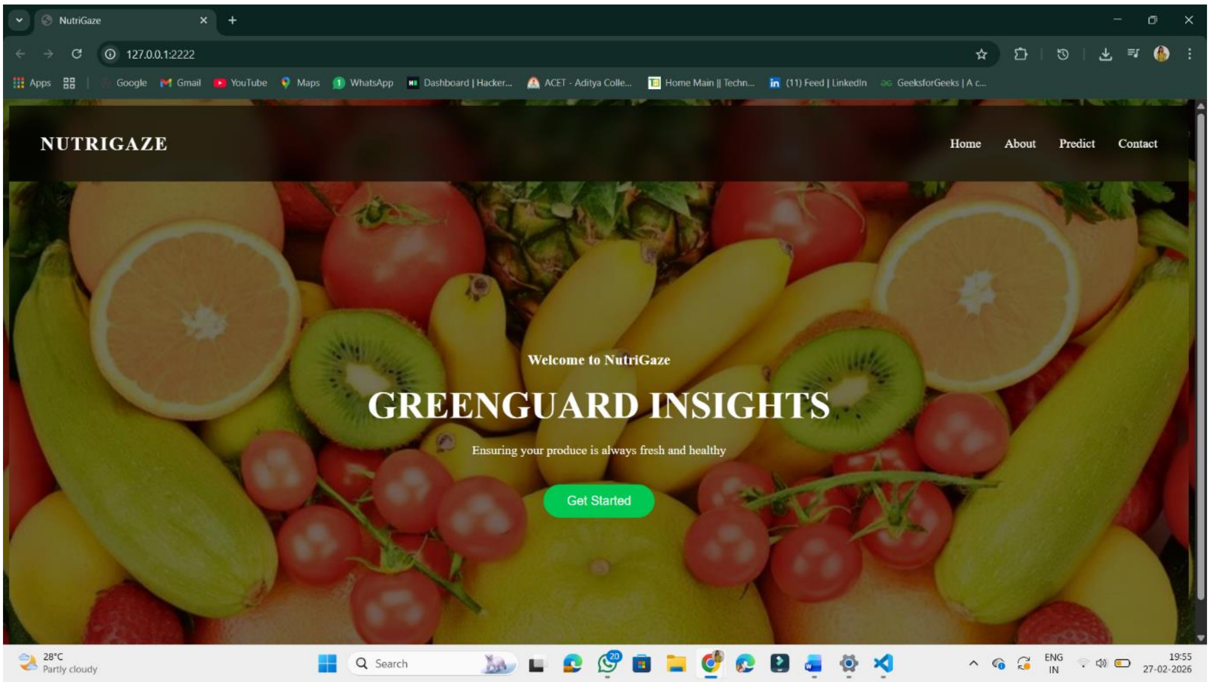
After running the command, the terminal generates a local host URL such as:
http://127.0.0.1:5000
The user can access the application by opening this URL in a web browser. Upon accessing the link, the web page loads successfully.



Step 3: Application Homepage Display

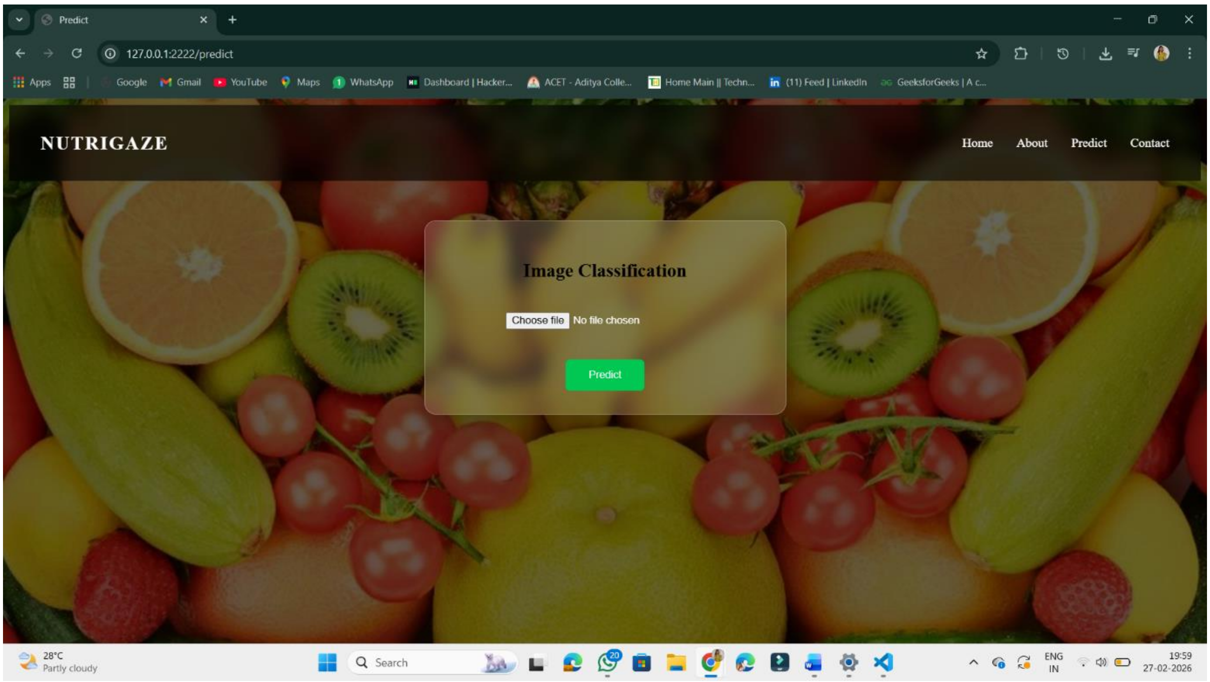
The Flask web page opens and displays the homepage of the **NutriGaze** application.

The interface is developed using Flask templates



Step 4: Dashboard Interaction

Once the dashboard loads, users can interact with the analysis whether the particular fruit or vegetable is fresh or rotten.



Step 5: Visualization Output

The dashboard displays the output by analysing it and predicting the result to the users.

Step 6: Deployment and Accessibility

After deployment, the application becomes accessible through a public URL. The system functions correctly online, and all interactive components operate without errors.

8. ADVANTAGES & DISADVANTAGES

Advantages

- Automated inspection
- Fast detection
- Reduced food waste
- User-friendly interface
- Real-time prediction

Disadvantages

- Depends on image quality
- Limited dataset
- Binary classification only

9. CONCLUSION

NutriGaze successfully demonstrates the practical implementation of deep learning and web technologies to solve a real-world problem in food quality inspection. The system integrates a trained Convolutional Neural Network (CNN) model with a Flask-based web application to automatically classify fruits and vegetables as fresh or rotten.

Through image preprocessing, model prediction, and real-time result display, NutriGaze provides an efficient and user-friendly solution for automated spoilage detection. The system reduces dependency on manual inspection, minimizes human error, and helps decrease food wastage in agriculture and retail sectors.

Performance testing shows that the application delivers fast response times, stable execution, and high classification accuracy. The web interface ensures ease of use, making the system accessible to farmers, retailers, and consumers.

Overall, NutriGaze proves that Artificial Intelligence and Computer Vision can significantly improve food quality monitoring processes. With further enhancements such as multi-class classification, cloud deployment, and real-time camera integration, the system has strong potential for large-scale industrial and commercial applications.

10. FUTURE SCOPE

The **NutriGaze** system provides a strong foundation for automated food quality inspection using deep learning. However, the project can be further enhanced and expanded in multiple ways to improve functionality, scalability, and real-world application.

1. Multi-Class Classification

Currently, the system classifies fruits and vegetables as either fresh or rotten. In the future, the model can be extended to support multi-class classification, such as:

- Identifying different types of fruits and vegetables
- Detecting multiple stages of ripeness
- Classifying different levels of spoilage

This will make the system more practical for commercial use.

2. Real-Time Camera Integration

The system can be upgraded to support real-time detection using a webcam or mobile camera. This will allow instant freshness detection without manually uploading images.

3. Mobile Application Development

A mobile app version of NutriGaze can be developed to enable farmers, retailers, and consumers to use the system directly from smartphones.

4. Cloud Deployment

The application can be deployed on cloud platforms such as AWS, Azure, or Google Cloud to allow large-scale access and industrial-level implementation.

5. IoT Integration

NutriGaze can be integrated with IoT devices in warehouses and storage facilities to monitor produce quality automatically.

6. Larger and Diverse Dataset Training

Improving the dataset size and diversity will increase model accuracy and robustness across different lighting conditions and environments.

7. AI-Based Quality Grading

Future versions can include grading systems (Grade A, B, C) instead of just Fresh/Rotten classification.

8. Supply Chain Monitoring System

The system can be integrated with supply chain management software to track product quality from farm to consumer.

11. APPENDIX

11.1 Source Code

The source code of the **NutriGaze** project includes

• Model training code
• Flask application code
• HTML templates
• CSS styling

11.2 Github &Project demo link

Github repository link:

Demo link: