DAYANANDA SAGAR COLLEGE OF ENGINEERING
(An Autonomous Institute affiliated to Visvesvaraya Technological University (VTU), Belagavi,
Approved by AICTE and UGC, Accredited by NAAC with 'A' grade & ISO 9001 – 2015 Certified Institution)
Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru-560 111, India

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING
(Accredited by NBA Tier 1: 2022-2025)

Mini Project (PROJ22IS66) report on

Packaged Food Label Analysis using Image Processing

Submitted in partial fulfillment of

Bachelor of Engineering
in
Information Science and Engineering

Submitted by

SHREYA DANDAPAT	**1DS22IS145**
SHREYA J	**1DS22IS147**
SHWETA KUMARI	**1DS22IS153**
TEJASWINI M B	**1DS22IS171**

Under the Guidance of
Mr. Yogesh B S
Assistant Professor, Dept. of ISE

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANASANGAMA,BELAGAVI-590018, KARNATAKA, INDIA
2024-25

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institute affiliated to Visvesvaraya Technological University (VTU), Belagavi,
Approved by AICTE and UGC, Accredited by NAAC with 'A' grade & ISO 9001 – 2015 Certified Institution)
Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru-560 111, India

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

(Accredited by NBA Tier 1: 2022-2025)

CERTIFICATE

This is to certify that the Mini Project report entitled **"Packaged Food Label Analysis using Image Processing"** carried out by **SHREYA DANDAPAT (1DS22IS145), SHREYA J (1DS22IS147), SHWETA KUMARI (1DS22IS153) and TEJASWINI M B (1DS22IS171),** in partial fulfillment for the **VI semester** of **Bachelor of Information Science and Engineering** of the Visvesvaraya Technological University, Belgaum, during the year 2024-2025. The Mini Project report has been approved as it satisfies the academic requirements prescribed for the Bachelor of Engineering degree.

Signature of the Guide	**Signature of the HoD**
Mr. Yogesh	**Dr. Annapurna P Patil**
Assistant Professor	**Dean Academics, Prof & Head**
Dept. of ISE, DSCE	**Dept. of ISE, DSCE, Bengaluru**
Bengaluru	

Name of the Examiners	**Signature with date**
1.	....
2.	....

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institute affiliated to Visvesvaraya Technological University (VTU), Belagavi,
Approved by AICTE and UGC, Accredited by NAAC with 'A' grade & ISO 9001 – 2015 Certified Institution)
Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru-560 111, India

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

(Accredited by NBA Tier 1: 2022-2025)

DECLARATION

We, **Shreya Dandapat (1DS22IS145), Shreya J (1DS22IS147), Shweta Kumari (1DS22IS153) and Tejaswini M B (1DS22IS171)** respectively, hereby declare that the mini project work entitled "**Packaged Food Label Analysis using Image Processing**" has been independently done by us under the guidance of **'Mr. Yogesh B S, Assistant Professor,** ISE department and submitted in partial fulfillment of the requirement for VI semester of the degree of **Bachelor of Information Science and Engineering** at **Dayananda Sagar College of Engineering**, an autonomous institution affiliated to VTU, Belagavi during the academic year 2024-25.

We hereby declare that the same has not been submitted in part or full for other academic purposes.

Shreya Dandapat	**1DS22IS145**
Shreya J	**1DS22IS147**
Shweta Kumari	**1DS22IS153**
Tejaswini M B	**1DS22IS171**

PLACE:

DATE:

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project.

We take this opportunity to express our sincere gratitude to **Dayananda Sagar College of Engineering** for having provided us with a great opportunity to pursue our Bachelor Degree in this institution.

In particular we would like to thank **Dr. B G Prasad,** Principal, Dayananda Sagar College of Engineering for his constant encouragement and advice.

Special thanks to **Dr. Annapurna P Patil,** Professor and HOD, Department of Information Science & Engineering, Dayananda Sagar College of Engineering for her motivation and invaluable support well through the development of this project.

We are highly indebted to our internal guide **Mr. Yogesh B S, Assistant Professor,** Department of Information Science & Engineering, Dayananda Sagar College of Engineering for their constant support and guidance. Our guide has been a great source of encouragement throughout the course of this mini project.

We express our sincere thanks to the Mini - Project Coordinators **Prof. Rekha Jayaram**, **Assistant Professor,** and **Prof. Vijetha**, **Assistant Professor** of the **Department of Information Science and Engineering** for their continuous support and guidance. We thank all teaching and non-teaching staff of the Department of Information Science and Engineering for their kind and constant support throughout the academic Journey.

Shreya Dandapat	**1DS22IS145**
Shreya J	**1DS22IS147**
Shweta Kumari	**1DS22IS153**
Tejaswini M B	**1DS22IS171**

ABSTRACT

This project presents an AI-powered system that automates the extraction and analysis of food additives from product labels using image processing and Optical Character Recognition (OCR). Leveraging Python libraries such as OpenCV and Tesseract, the system scans food label images to detect listed ingredients and additive codes (e.g., E-numbers, INS codes). These extracted additives are then cross-referenced against a curated database of food chemicals, including known health risks. The application, built with user-friendly interfaces in Streamlit and Gradio, not only identifies potentially harmful preservatives and artificial colors but also retrieves real-time health-related information from the web. This tool empowers consumers to make informed decisions about their dietary choices by making food label transparency accessible and interactive.

Keywords: *Optical Character Recognition, Food Additive Detection, Image Processing, Ingredient Risk Analysis, Consumer Health Awareness*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Sl. No	Abbreviation	Full Description
1.	OCR	Optical Character Recognition
2.	NLP	Natural Language Processing
3.	UI	User Interface

1. INTRODUCTION

1.1 Overview

In today's fast-paced world, packaged food consumption has skyrocketed due to convenience and accessibility. However, the lack of transparency and understanding around food ingredient labels poses significant health risks to consumers. Many individuals struggle to interpret complex ingredient lists, which often contain food colorants, chemical additives, and allergens that could be harmful if consumed in excess. To address this, the Packaged Food Label Scanner is a web-based application that uses OCR and NLP techniques to extract, analyze, and assess the potential risks of ingredients in packaged foods. This scanner not only alerts users about harmful substances but also recommends safer alternatives and supports user personalization through dietary preferences and allergen tracking.

1.2 Problem Statement

Consumers lack an accessible and accurate way to evaluate the safety of packaged food ingredients. Most ingredient lists are written in technical or ambiguous language, often making it difficult to recognize allergens, artificial additives, or chemicals that may exceed the Daily Recommended Intake (DRI). This lack of awareness can result in health risks, especially for individuals with allergies, chronic illnesses, or specific dietary restrictions. The need for a reliable, real-time solution to scan, identify, and analyze these ingredients has become increasingly critical.

1.3 Objectives

The main objectives of the Packaged Food Ingredient Scanner are:

- To extract ingredient lists from images using high-accuracy Optical Character Recognition.
- To identify and classify food components, especially allergens and additives, using Machine Learning and regulatory data.
- To analyze ingredients against safety thresholds (DRI) and display health risks.
- To ensure real-time performance, data privacy, and integration with external APIs and databases for updated food information.

1.4 Motivation

- Health Risks from Packaged Foods: With increasing chronic conditions like obesity, diabetes, and allergies, users must be informed about what they consume.
- Lack of Consumer Awareness: Most people do not understand technical ingredient labels and the potential impact of chemical additives.
- Need for Instant Risk Assessment: Modern consumers need mobile solutions that deliver immediate, accurate, and personalized food risk analysis.
- Tech for Social Good: Leveraging OCR and AI for public health awareness contributes to building a more informed and health-conscious society.

1.5 Hardware and Software Requirements Specification

Hardware Requirements:

- Smartphone Camera or External Scanner – Capturing food label images for OCR processing.
- Wireless Communication Modules – Wi-Fi / 4G for cloud-based ingredient analysis.

Software Requirements:

- Server Frameworks: Node.js, Python, Jupyter Notebook
- Database: Firebase / SQL (MySQL, PostgreSQL) for structured data storage.
- Cloud Services: Firebase for user authentication and storage.
- Web Dashboard: Streamlit for quick deployment of web-based ingredient analysis.
- OCR & Image Preprocessing: Python3, Pytesseract, Tesseract-OCR for text extraction, OpenCV for image enhancement (noise reduction, skew correction, resizing).
- Natural Language Processing (NLP): Regular Expressions (re module) for detecting hidden allergens, Text tokenization & Named Entity Recognition (NER) for ingredient classification.

1.6 Project Budget Plan

Expenditure	**Budget**	**Actual**
Training / Online courses	5,000	
Materials and supplied	5,000	
Software	1,000	
Hardware	-	
Others	10,000 (OCR camera)	
Paper Presentation/ Submission	8,000	
Proposal Submission	200	
Total	17,200	

2. LITERATURE SURVEY

Sl. No.	Authors / Year of Publication	Title of Article	Methods Used	Results	Remarks
01	Rohini B., Divya Pavuluri, L.S. Kumar, V. Soorya, Aravinth Jagatheesan, 2024	NutrifyAI: An AI-Powered System for Real-Time Food Detection and Nutritional Analysis	Combined OCR with deep learning for identifying allergens and nutrients on food packaging.	Enhanced consumer safety by providing detailed information about food products.	Limited to packaged foods; may not generalize to loose or natural foods.
02	Chinyelu Maureen Uzoma, Nosakhare J. Uwugiaren, Chike A. Ugwunze, Hauwa M. Umaru, 2024	Automating Nutritional Claim Verification: The Role of OCR and Machine Learning in Enhancing Food Label Transparency	Assessed the implications of food labeling policies using OCR and machine learning.	Identified potential of OCR and ML in improving consumer understanding of food labels.	Emphasized the need for clearer label formats and technological interventions.
03	Yaksh Shah, Nandan Jariwala, Bhakti Kachhia, Prachi Shah, 2023	Delving Deep into NutriScan: Automated Nutrition Table Extraction and Ingredient Recognition	Utilized EfficientDet for object detection and PaddleOCR for text extraction; employed regular expressions for parsing nutritional information.	Successfully extracted detailed nutritional information, including salt, fat, and protein content, from packaged food labels.	Provided an end-to-end solution for automating the extraction of nutritional data from food packaging.
04	Mark D. Miller et al., 2022	Potential impacts of synthetic food dyes on activity and attention in	Systematic review of 27 clinical trials and animal studies	64% of challenge studies showed positive associations;	Suggests current FDA ADIs may not adequately protect

		children: a review of the human and animal evidence		52% statistically significant	children's neurobehavioral health
05	Tejashree A. More, Zoya Shaikh, Ahmad Ali, 2021	Artificial Sweeteners and their Health Implications: A Review	Literature review on artificial sweeteners' characteristics and health effects	Highlights potential health risks ranging from headaches to cancer	Emphasizes need for long-term studies to evaluate safety
06	Jiangpeng He et al., 2021	An End-to-End Food Image Analysis System	Deep learning framework integrating food localization, classification, and portion size estimation	Demonstrated effectiveness in real-life food image datasets	Plans to integrate system into mobile and cloud platforms for real-time dietary monitoring
07	Anuj Khasgiwala, 2018	Word Recognition in Nutrition Labels with Convolutional Neural Network	Developed a Convolutional Neural Network (CNN) model for recognizing words in nutrition labels.	Improved the accuracy of text recognition in nutrition labels, facilitating better dietary assessments.	Highlighted the potential of CNNs in processing and interpreting nutritional information from food packaging.
08	Apoorva P., 2016	Nutrition Facts Label Processing using Image Segmentation and Token Matching based on OCR	Combined Otsu's method for segmentation with Tesseract OCR to identify and tabulate line items on nutrition facts labels	Achieved better success rate in identifying label line items compared to unprocessed images	Future work includes refining segmentation techniques and improving OCR accuracy for diverse label designs

3. PROBLEM ANALYSIS & DESIGN

3.1 Existing System

Currently, most food label analyzers in use provide limited functionality. These systems are typically constrained in the following ways:

- Limited OCR Accuracy: Existing solutions often depend on basic OCR engines that struggle with blurry images, unusual fonts, or poor lighting conditions.
- Lack of Ingredient Classification: While text may be extracted, these systems do not classify ingredients based on user-specific dietary restrictions or known health risks.
- No Customization or Personalization: Most systems do not allow users to set preferences such as allergens, dietary needs (e.g., vegan, gluten-free), or banned ingredients.
- Minimal User Interaction: The user interface in these systems is often rigid, without feedback mechanisms or the ability to save scanned results.
- Absence of API Integration: Many systems do not utilize nutrition or health databases (like Edamam or Open Food Facts) to enrich the scanned data.
- No Real-Time Validation: There is no backend validation or intelligent matching of scanned ingredients with standardized ingredient databases or regulatory lists.

These limitations highlight the need for a more intelligent, user-focused system capable of not just reading, but interpreting, evaluating, and recommending safer alternatives based on ingredient labels.

3.2 Proposed System

Based on the earlier-pasted code, the proposed system is a streamlined application that allows users to scan food ingredient lists using a camera or file upload interface. It extracts text using OCR and checks whether any of the listed ingredients are among a set of harmful or undesired substances.

1. **OCR-Based Ingredient Extraction**:
    - Accepts image input (e.g., food package label).
    - Uses OCR (Google Vision) to convert images to text.
    - Identifies individual ingredients by splitting the OCR output using delimiters like commas.

2. **Ingredient Verification Logic**:
    - Compares each extracted ingredient against a predefined list of harmful ingredients (e.g., "sodium nitrate", "MSG", etc.).
    - If any harmful ingredient is detected, it is highlighted and added to a separate list.

3. **Frontend Presentation**:
    - Built in Streamlit and Gradio.
    - Displays ingredients extracted from the image and harmful ingredients.

4. **User Interaction**:
    - Provides an input field and button for OCR-triggered scanning.
    - Displays processed output clearly with visual alerts for flagged items.

5. **Simple Workflow**:
    - Ideal for mobile or desktop web apps.
    - Lightweight architecture that can be integrated with future ML models or databases.

This system represents a foundational version of a larger food-label safety analyzer — focused, fast, and ready for enhancement with more complex integrations such as NLP, regulatory compliance, and dietary customizations.

3.3 Identified Tools/Libraries/Software

Frontend: Streamlit, React Native

Backend: Node.js, Python

OCR: Google Vision API, Tesseract OCR

Database: Excel, Firebase

APIs: Regulatory Databases (FDA, FSSAI, EFSA)

Libraries: Beautiful Soup, Regular Expressions, OpenCV, Pandas

3.4 Architectural Block Diagram and Corresponding System Modeling

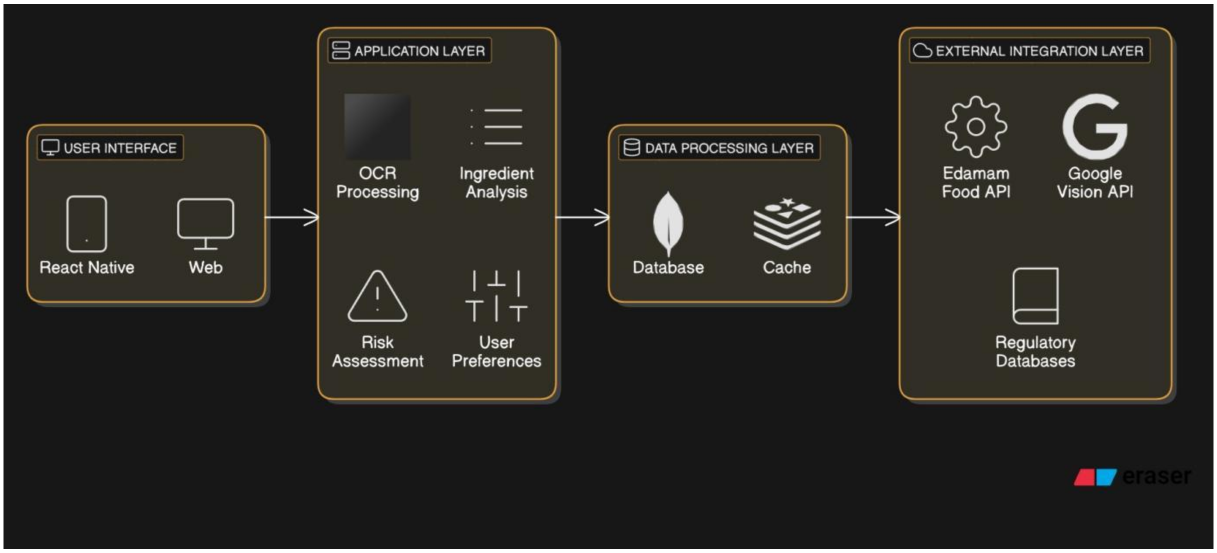


Fig 1: Architectural Model

1. User Interface Layer

The frontend includes a web interface that enables manual image uploads and displays ingredient analysis results. This layer interacts with the backend for OCR processing and ingredient analysis.

2. Application Layer

This layer is responsible for the core business logic and consists of:

- OCR Processing – Extracts text from food packaging images.
- Ingredient Analysis – Identifies and categorizes ingredients from the extracted text.
- Risk Assessment – Maps ingredients to potential health risks.
- User Preferences – Stores dietary restrictions and preferences for personalized analysis.

The application layer communicates with both the data processing layer and external APIs for accurate ingredient validation.

3. Data Processing Layer: This layer ensures efficient storage and retrieval of ingredient-related data. The database stores user preferences, scanned data, and ingredient classifications.

4. External Integration Layer

The system integrates with third-party services to enhance ingredient validation and OCR accuracy:

- Google Vision API (OCR) – Enhances text extraction from food labels.
- Regulatory Databases (FDA, FSSAI, EFSA) – Ensures compliance with food safety regulations.

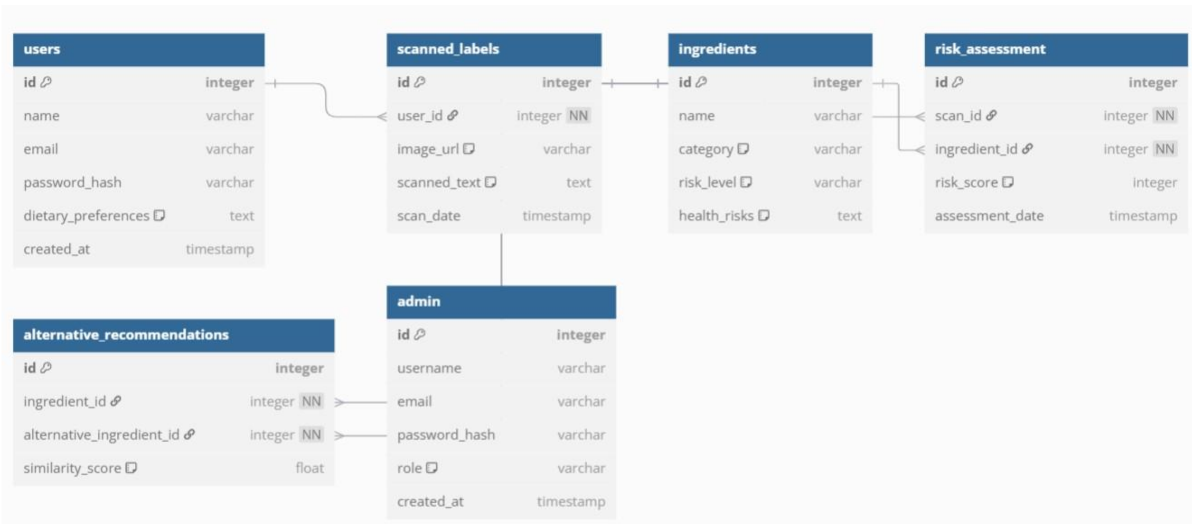


Fig 2: Entity Relationship Model

Entity-Relationship (ER) Model:

- Entities:
    - Users: User ID, Name, Email, Password, Dietary Preferences
    - Scanned Labels: Scan ID, User ID, Image URL, Extracted Text, Scan Timestamp
    - Ingredients: Ingredient ID, Name, Category, Risk Level, Health Risks
    - Risk Assessment: Assessment ID, Scan ID, Ingredient ID, Risk Score, Assessment Timestamp
    - Alternative Recommendations: Ingredient ID, Alternative Ingredient ID, Similarity Score
    - Admin: Admin ID, Username, Email, Password, Role

- Relationships:
    - A user can perform multiple scanned label analyses (one-to-many).
    - A scanned label can contain multiple ingredients (many-to-many).

- An ingredient can have multiple alternative recommendations (one-to-many).
- A risk assessment is linked to both a scanned label and ingredients (many-to-one).
- Admins oversee risk assessments and ingredient safety data.

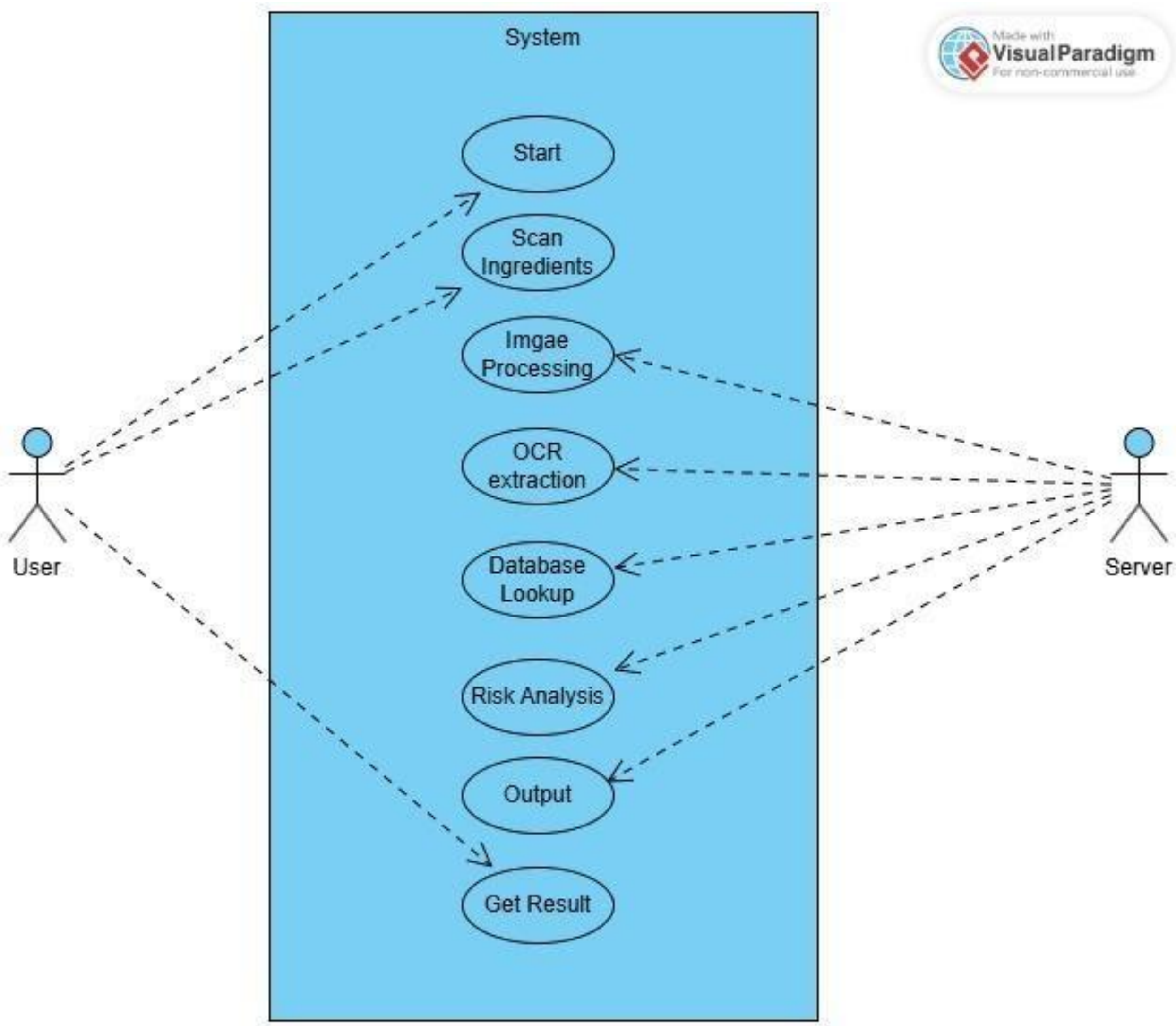


Fig 3: Use Case Diagram

Actors:

1. User: Can scan food labels, view analysis reports, receive recommendations, and provide feedback.
2. Admin: Manages the ingredient database, updates risk levels, verifies OCR accuracy, and oversees system performance.

Use Cases:

- User Scans Label: The user uploads an image of a food label to the system.
- OCR Extracts Ingredients: The system extracts text from the image and identifies ingredients.

- System Analyzes Ingredients: The system checks each ingredient against the database for risk assessment.
- Risk Report Generation: The system compiles the ingredient risk analysis into a report.
- Alternative Recommendation: If high-risk ingredients are detected, the system suggests safer alternatives.
- Admin Manages Database: The admin updates ingredient information, adjusts risk levels, and ensures accuracy.

Data Flow Process:

1. User Inputs Data: The process starts when a user uploads a food label image to the system.
2. OCR Processing: The system extracts text from the uploaded image using Optical Character Recognition (OCR).
3. Ingredient Matching: Extracted ingredients are compared against the database to determine their category and associated health risks.
4. Risk Assessment: Based on predefined risk levels, the system assigns a risk score to each ingredient found in the scanned label.
5. Recommendation Generation: If high-risk ingredients are detected, the system suggests alternative ingredients.
6. Data Storage: All assessments are stored in the database for future reference and system improvement.

Fig 4: Data Flow Diagram

4. IMPLEMENTATION

4.1 Overview of System Implementation

The system is designed to identify food additives from product label images and assess their associated health risks. It integrates **image processing**, **Optical Character Recognition (OCR)**, **data matching** from an Excel database, and **web scraping** using Google Search and BeautifulSoup. The system is deployed with a **Tkinter GUI** to allow user interaction and displays extracted data and health risks in a user-friendly format.

Key technologies used:

- Python
- OpenCV for image preprocessing
- Tesseract OCR for text extraction
- pandas for database handling
- fuzzywuzzy for string matching
- requests & BeautifulSoup for scraping
- tkinter for the GUI interface

4.2 Module Description

Module 1: Image-to-Web Search Additive Health Risk Checker

1. Tesseract Configuration

    - Sets the path to the local Tesseract-OCR executable to enable OCR (Optical Character Recognition).

2. Ingredient Extraction from Image

    - Reads an image using OpenCV.
    - Converts it to grayscale.
    - Uses `pytesseract` to extract text.
    - Applies regex to isolate the “Ingredients” section.
    - Cleans and splits the ingredient string into a list using commas, newlines, and semicolons.

3. Web Search for Health Risks

- For each ingredient, generates a search query like “XYZ food additive health risks.”
- Uses `googlesearch` to fetch top 5 links.
- Parses each URL using `requests` and `BeautifulSoup`.
- Extracts the first relevant paragraph that mentions the ingredient and health.

4. GUI using Tkinter

- GUI window allows users to upload an image.
- Displays extracted ingredients and health information from online sources.
- Includes scrollable text area to view results interactively.

Module 2: OCR + Fuzzy Match with Excel Database

1. Excel File Handling

- Lets the user select an Excel file via file dialog (predefined path in your version).
- Reads the file into a Pandas DataFrame.
- Cleans additive names and codes to lowercase strings for reliable matching.

2. Image File Input & OCR

- Lets the user select an image containing ingredients.
- Converts the image to grayscale.
- Extracts text using `pytesseract`.

3. Code Normalization & Extraction

- Uses regex to extract additive codes (like "330", "150d", "160c", "INS 102", etc.).
- Normalizes different code formats to a standard structure.

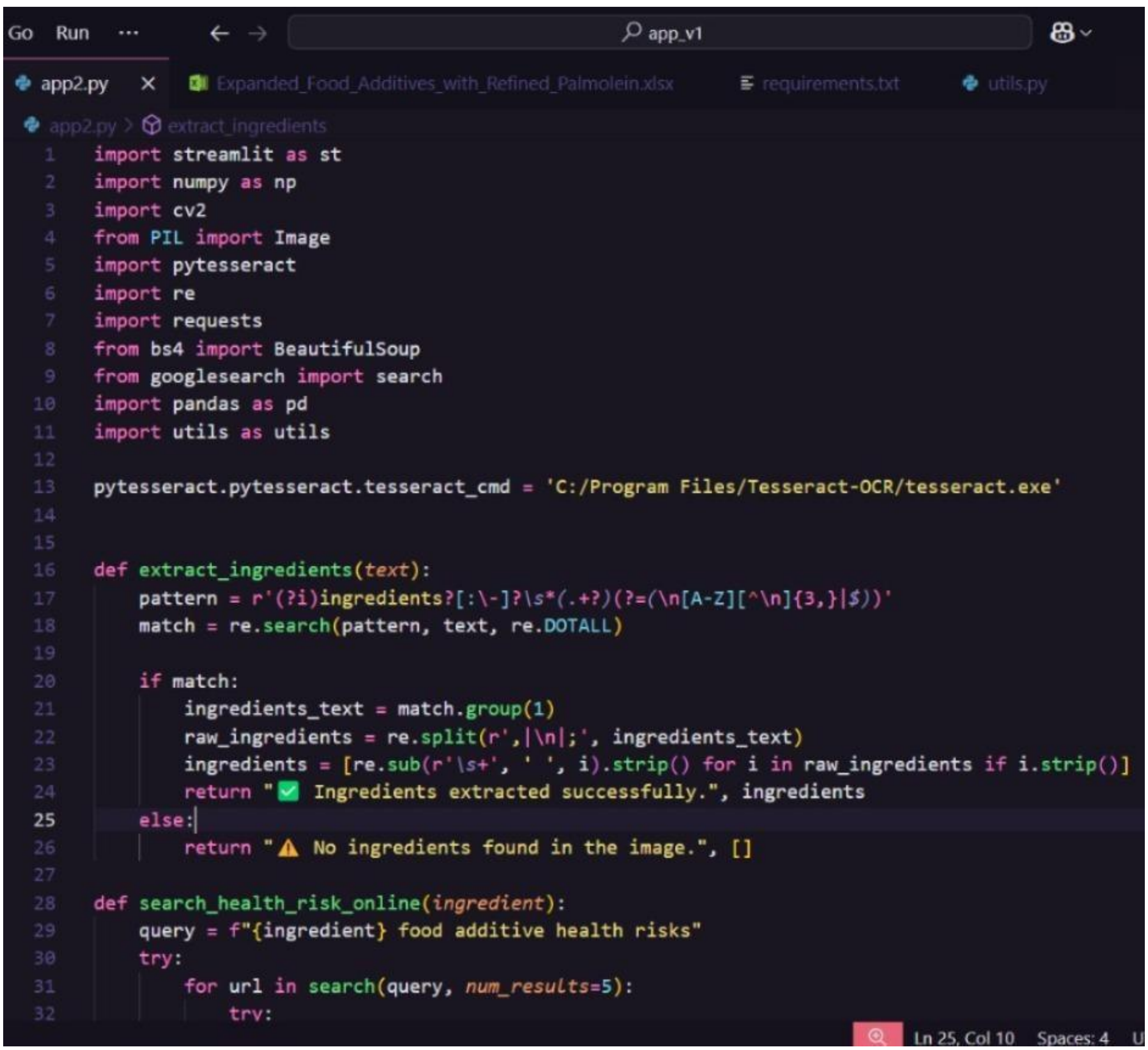
4. Fuzzy Matching Logic

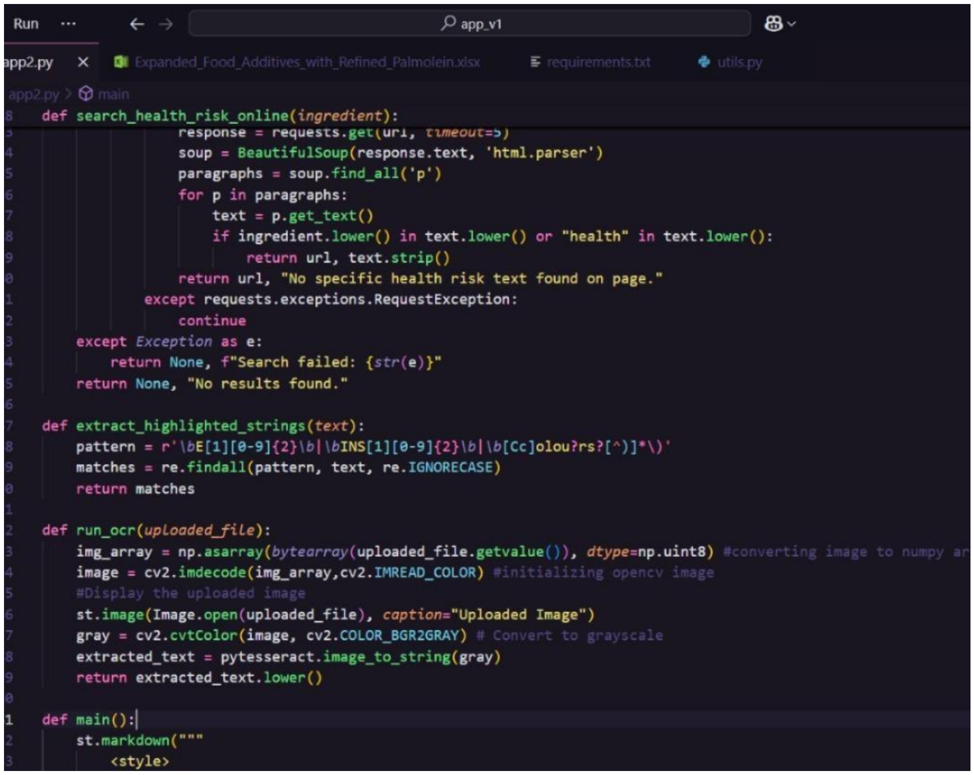
- Compares OCR-extracted text against additive names and INS codes using `fuzzywuzzy`.
- Matches additives from the image to entries in the Excel dataset based on a similarity threshold.
- Collects matched entries with associated health risks.

5. Console Output

- Displays matched additive names, codes, and associated health risks in the terminal.

4.3 Code Snippets



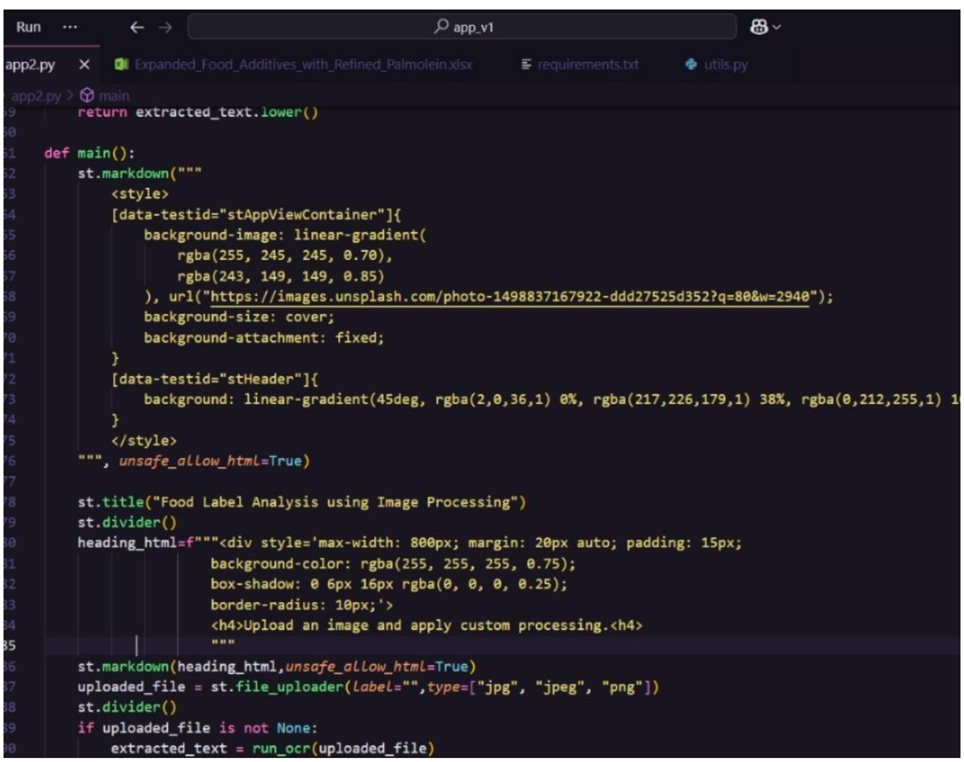


```python
def search_health_risk_online(ingredient):
                response = requests.get(url, timeout=5)
                soup = BeautifulSoup(response.text, 'html.parser')
                paragraphs = soup.find_all('p')
                for p in paragraphs:
                    text = p.get_text()
                    if ingredient.lower() in text.lower() or "health" in text.lower():
                        return url, text.strip()
                return url, "No specific health risk text found on page."
            except requests.exceptions.RequestException:
                continue
    except Exception as e:
        return None, f"Search failed: {str(e)}"
    return None, "No results found."

def extract_highlighted_strings(text):
    pattern = r'\bE[1][0-9]{2}\b|\bINS[1][0-9]{2}\b|\b[Cc]olou?rs?[^)]*\)'
    matches = re.findall(pattern, text, re.IGNORECASE)
    return matches

def run_ocr(uploaded_file):
    img_array = np.asarray(bytearray(uploaded_file.getvalue()), dtype=np.uint8) #converting image to numpy ar
    image = cv2.imdecode(img_array,cv2.IMREAD_COLOR) #initializing opencv image
    #Display the uploaded image
    st.image(Image.open(uploaded_file), caption="Uploaded Image")
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY) # Convert to grayscale
    extracted_text = pytesseract.image_to_string(gray)
    return extracted_text.lower()

def main():
    st.markdown("""
        <style>
```



```python
    return extracted_text.lower()

def main():
    st.markdown("""
        <style>
        [data-testid="stAppViewContainer"]{
            background-image: linear-gradient(
                rgba(255, 245, 245, 0.70),
                rgba(243, 149, 149, 0.85)
            ), url("https://images.unsplash.com/photo-1498837167922-ddd27525d352?q=80&w=2940");
            background-size: cover;
            background-attachment: fixed;
        }
        [data-testid="stHeader"]{
            background: linear-gradient(45deg, rgba(2,0,36,1) 0%, rgba(217,226,179,1) 38%, rgba(0,212,255,1) 1
        }
        </style>
    """, unsafe_allow_html=True)

    st.title("Food Label Analysis using Image Processing")
    st.divider()
    heading_html=f"""<div style='max-width: 800px; margin: 20px auto; padding: 15px;
                    background-color: rgba(255, 255, 255, 0.75);
                    box-shadow: 0 6px 16px rgba(0, 0, 0, 0.25);
                    border-radius: 10px;'>
                    <h4>Upload an image and apply custom processing.<h4>
                    """
    st.markdown(heading_html,unsafe_allow_html=True)
    uploaded_file = st.file_uploader(label="",type=["jpg", "jpeg", "png"])
    st.divider()
    if uploaded_file is not None:
        extracted_text = run_ocr(uploaded_file)
```

Fig 5: Code Snippets

5. TESTING

Software testing is a crucial phase in the software development life cycle that helps ensure the application behaves as expected under various conditions. It validates both functionality and performance, reducing the risk of failures in production. Testing enhances product quality, user satisfaction, and system stability.Various testing approaches like unit, integration, and system testing target specific aspects of the software. Incorporating automated testing in CI/CD pipelines enables faster development cycles, immediate feedback, and efficient detection of bugs, ultimately supporting robust and maintainable software delivery.

White Box Testing

White box testing, also known as structural testing, involves examining the internal code structure to verify correct behavior. In this project, it was used to test core functions like extract_ingredients, extract_highlighted_strings, run_ocr, and search_health_risk_online. By analyzing code paths and edge cases, this approach helped identify logic errors and inefficiencies, ensuring each component works reliably and contributes to overall system stability.

Unit Testing

Unit testing is a fundamental aspect of white box testing, focusing on verifying the correctness of individual units of code, typically functions or methods. For this project, unit tests were developed for all major utility functions. For example, the extract_ingredients function was tested with a variety of input strings to confirm that it correctly identifies and parses ingredients from text extracted via OCR. The extract_highlighted_strings function was tested to ensure it reliably detects food additive codes and color additives in different formats. The run_ocr function was validated using both valid and corrupted image files to check its robustness and error handling. Additionally, the search_health_risk_online function was tested with both valid and invalid additive names, as well as simulated network failures. These unit tests were executed using automated testing frameworks, and both expected and unexpected outputs were compared to actual results to identify and fix defects early in the development process. This systematic unit testing approach has greatly enhanced the accuracy, stability, and quality assurance of the food label analysis system.

Test Case ID	Test Case Description	Input	Expected Output	Actual Output	Test Result
UT001	Extract ingredients from text with ingredients section	"Ingredients: water, E102, INS 110"	("Ingredients extracted successfully.", ["water", "E102", "INS 110"])	As expected	Pass
UT002	No ingredients section in text	"Nutrition Facts: Energy 200kcal"	("No ingredients found in the image.", [])	As expected	Pass
UT003	Extract highlighted additive codes from text	"Contains E102, INS 110, Colours (sunset yellow)"	["E102", "INS 110", "Colours (sunset yellow)"]	As expected	Pass
UT004	Extract highlighted codes from text with no matches	"Contains vitamins and minerals"	[]	As expected	Pass
UT005	Search health risk online (valid ingredient, mocked)	"E102"	Tuple with a URL containing "E102" and a paragraph mentioning "E102" or health risk	As expected (when mocked)	Pass
UT006	Search health risk online (invalid ingredient, mocked)	"NonexistentAdditive123"	(None, "No results found.") or similar error message	As expected (when mocked)	Pass
UT007	run_ocr returns lowercased text from image (mocked)	Mocked image file with text "SUGAR, E102, WATER"	"sugar, e102, water"	As expected (when mocked)	Pass

Fig. 6 Unit Testing

Test Case ID	Test Case Description	Input	Expected Output	Actual Output	Test Result
UT008	Extract ingredients with malformed delimiter	"Ingredients water E102 INS 110"	("Ingredients extracted successfully.", ["water E102 INS 110"])	("No ingredients found in the image.", [])	Fail
UT009	Extract highlighted codes with lowercase 'e'	"contains e102, ins110, colours (sunset yellow)"	["e102", "ins110", "colours (sunset yellow)"]	["ins110", "colours (sunset yellow)"]	Fail
UT010	run_ocr with corrupted image file (mocked)	Corrupted image file	Raises exception or returns empty string	Function crashes with error	Fail
UT011	search_health_risk_online with network failure (mocked)	"E102" (simulate network timeout)	(None, "Search failed: <error message>")	(None, "Search failed: <error message>")	Pass
UT012	extract_ingredients with empty string	""	("No ingredients found in the image.", [])	As expected	Pass

Fig. 6 Unit Testing

Explanation of Failed Cases:

UT008: The function expects a delimiter (comma, semicolon, or newline) after "Ingredients" and fails when none is present.

UT009: The regex pattern does not match lowercase 'e' in "e102", missing some codes.

UT010: The OCR function does not handle corrupted files gracefully and may crash

Test Report

Number of test cases executed - 12

Number of test cases passed - 9

Number of test cases failed - 3

Blackbox Testing

Black-box testing is a software testing methodology where the tester evaluates the application's functionality without any knowledge of its internal code structure or implementation details. The primary goal is to ensure that the software meets its specified requirements by providing various inputs and verifying that the outputs are as expected. This approach simulates the perspective of an end user, focusing on how the system behaves in response to different scenarios, including both typical and edge cases. Black-box testing helps identify functional discrepancies and ensures that the application delivers correct and reliable results.

Integration Testing

Integration testing is a key component of black-box testing used in this project. It involves testing the interactions between different modules of the system to verify that they work together as intended. In the context of the food label analysis project, integration testing was performed by combining modules such as image upload and OCR processing, ingredient extraction, additive identification, and health risk analysis. For example, an integration test would involve uploading a food label image, extracting the text, parsing the ingredients, identifying additives, and retrieving health risk information from both online and local sources. The tests were conducted without reference to the internal logic of each module, focusing instead on the accuracy and reliability of the overall workflow. This ensures that all components interact seamlessly and that the system produces consistent and accurate results for end users.

Test Case ID	Test Case Description	Input	Expected Output	Actual Output	Test Result
TC001	OCR extracts text from uploaded label image	Clear image of Nutella label (JPG)	Extracted text contains "hazelnuts", "skim milk"	Extracted text contains "hazelnuts", "skim milk"	Pass
TC002	Ingredient extraction from OCR text	"Ingredients: water, E102, INS 110"	Status: "Ingredients extracted successfully." Ingredients: ["water", "E102", "INS 110"]	Status: "Ingredients extracted successfully." Ingredients: ["water", "E102", "INS 110"]	Pass
TC003	No ingredient section present	"Nutrition Facts: Energy 200kcal"	Status: "No ingredients found in the image." Ingredients: []	Status: "No ingredients found in the image." Ingredients: []	Pass
TC004	Extract highlighted additive codes from text	"Contains E102, INS 110, Colours (sunset yellow)"	["E102", "INS 110", "Colours (sunset yellow)"]	["E102", "INS 110", "Colours (sunset yellow)"]	Pass
TC005	Online health risk search for additive	Ingredient: "E102"	Returns URL and paragraph mentioning "E102" and health risk info	Returns relevant URL and paragraph	Pass
TC006	Local database matching for additive	Extracted text: "E102" DB contains E102	Displays matched additive "E102" with health risk and ADI value	Displays matched additive with risk and ADI	Pass
TC007	Handle missing Excel database file	No Excel file at path	Error message: "No Excel file selected." or graceful handling	Error message shown	Pass
TC008	Handle corrupted image upload	Corrupted image file	Displays error or message indicating image cannot be processed	Displays error message	Pass

Fig. 7 Integration Testing

6. RESULTS

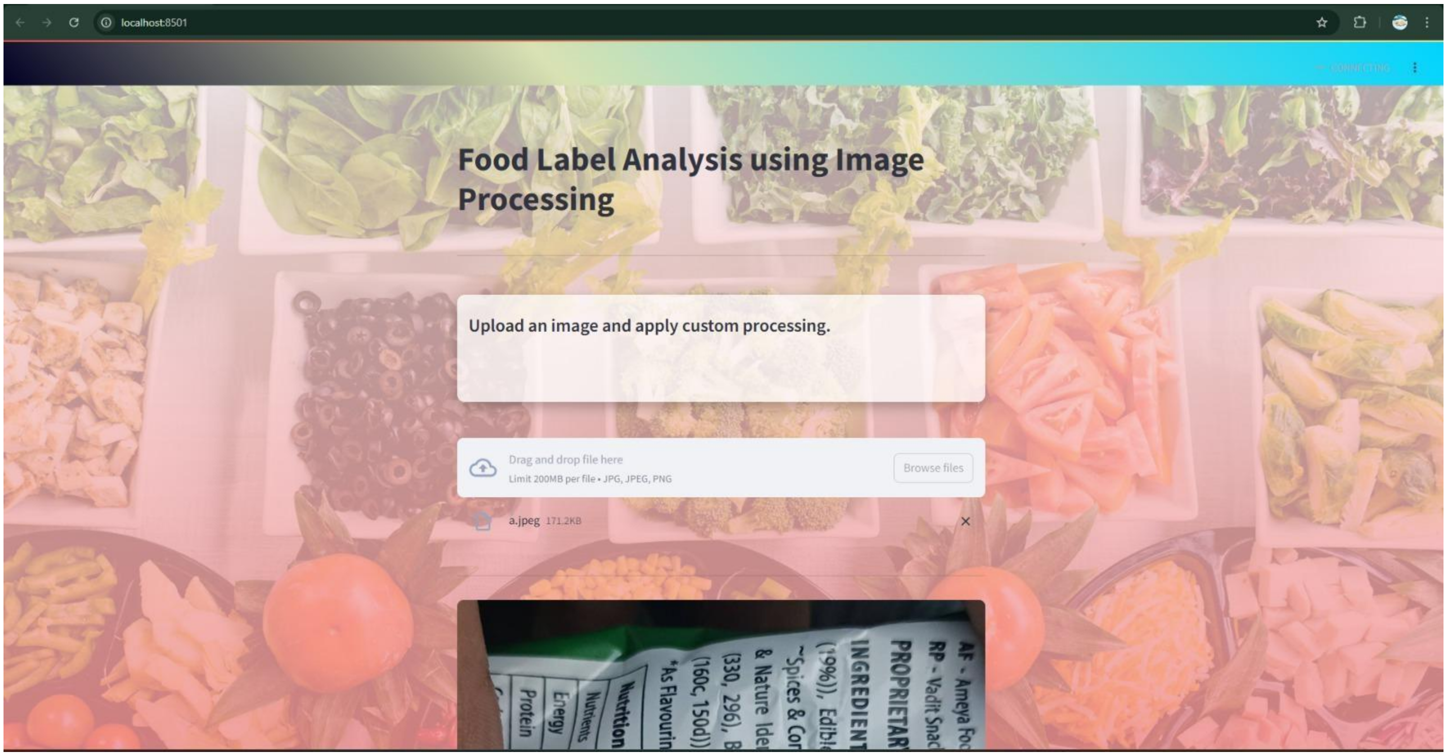


Fig 6: Image Input

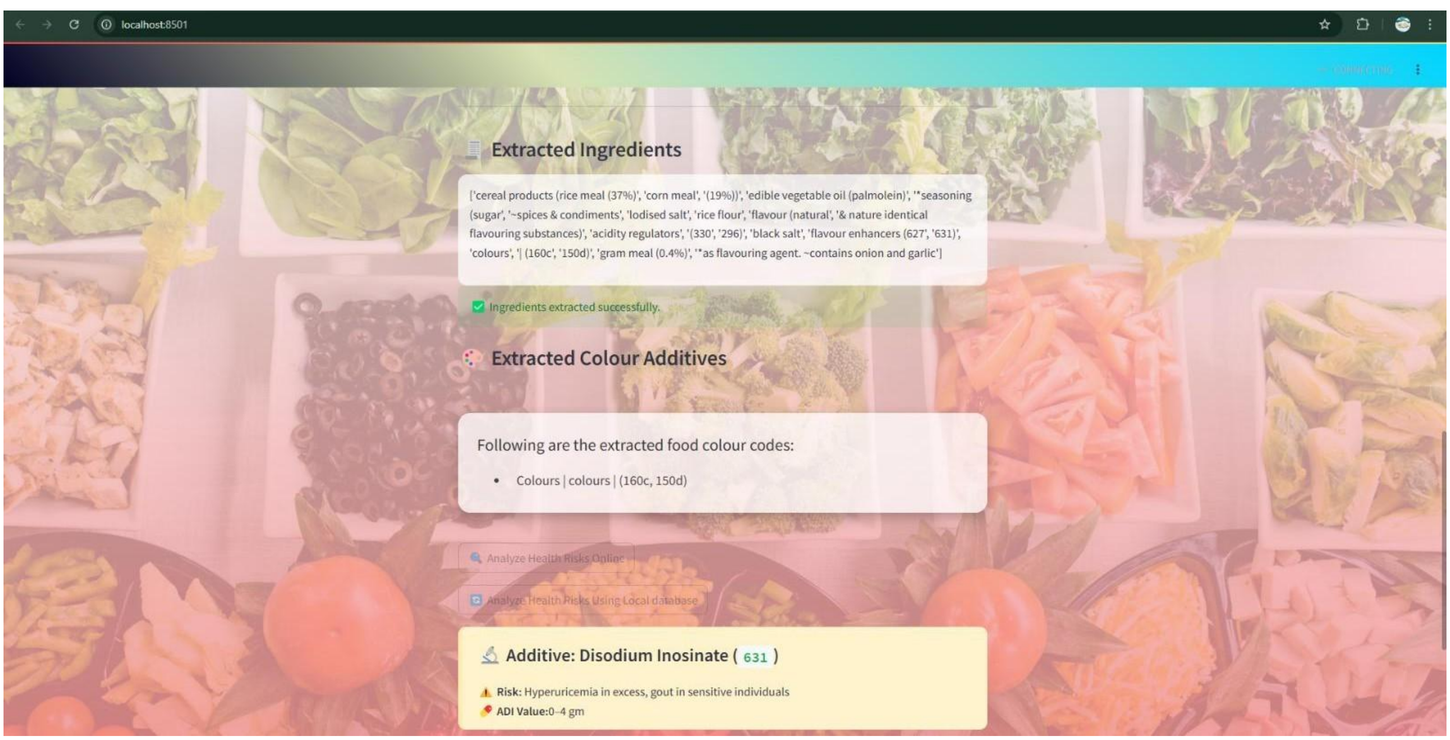


Fig 7: Extracted Details and Insights

5. CONCLUSION AND FUTURE SCOPE

This project explores the application of Optical Character Recognition (OCR) and image processing techniques to extract and interpret nutritional information from packaged food labels. Through a comparative review of eight key studies, it is evident that traditional computer vision methods—such as Otsu's thresholding, region segmentation, and OCR engines like Tesseract—can effectively identify and structure nutrition facts from various label formats. These approaches have improved the accessibility and transparency of nutritional data for consumers, especially in settings where AI or deep learning infrastructure is not feasible. The project highlights that even without advanced AI/ML techniques, accurate and practical systems can be built to aid consumers in making informed dietary choices.

Future Scope

1. **Support for Diverse Label Formats**
   Improve OCR accuracy by incorporating adaptive image pre-processing techniques (e.g., skew correction, contrast enhancement) to handle non-standard or poor-quality labels.

2. **Multi-language OCR Capabilities**
   Extend support to recognize regional and international languages on nutrition labels using open-source OCR tools like Tesseract's language training data.

3. **Real-Time Mobile App Integration**
   Deploy the solution as a lightweight mobile or web app that allows users to scan food labels on the go without internet dependency or heavy processing.

4. **Barcode-Based Supplementation**
   Add barcode scanning functionality to cross-check and fetch standardized nutrition data from open databases like Open Food Facts to complement OCR output.

5. **Label Comparison Tool**
   Create a feature to compare multiple nutrition labels side-by-side, assisting users in selecting healthier options.

6. **Offline Functionality**

   Optimize the system for offline usage so that users can access label analysis in remote or low-connectivity areas, improving accessibility.

7. **Incorporation into Assistive Tech**

   Adapt the tool for visually impaired users by converting OCR output into speech or using large-font display options for better readability.

REFERENCES

1. R. B. Rohini, D. Pavuluri, L. S. Kumar, V. Soorya, and A. Jagatheesan, "NutrifyAI: An AI-Powered System for Real-Time Food Detection and Nutritional Analysis," *arXiv preprint arXiv:2408.10532*, 2024. [Online]. Available: https://arxiv.org/html/2408.10532v2

2. C. M. Uzoma, N. J. Uwugiaren, C. A. Ugwunze, and H. M. Umaru, "Automating Nutritional Claim Verification: The Role of OCR and Machine Learning in Enhancing Food Label Transparency," *ResearchGate*, 2024. [Online]. Available: https://www.researchgate.net/publication/387921937

3. Y. Shah, N. Jariwala, B. Kachhia, and P. Shah, "Delving Deep into NutriScan: Automated Nutrition Table Extraction and Ingredient Recognition," *International Journal for Research in Applied Science & Engineering Technology*, vol. 11, no. 11, Nov. 2023. [Online]. Available: https://www.ijraset.com/research-paper/delving-deep-into-nutriscan-automated-nutrition-table-extractionIJRASET+1IJRASET+1

4. M. D. Miller *et al.*, "Potential impacts of synthetic food dyes on activity and attention in children: a review of the human and animal evidence," *Environmental Health*, vol. 21, no. 1, pp. 1–17, 2022. [Online]. Available: https://ehjournal.biomedcentral.com/articles/10.1186/s12940-022-00849-9BioMed Central

5. T. A. More, Z. Shaikh, and A. Ali, "Artificial Sweeteners and their Health Implications: A Review," *Biosciences Biotechnology Research Asia*, vol. 18, no. 2, pp. 1–9, 2021. [Online]. Available: https://www.biotech-asia.org/vol18no2/artificial-sweeteners-and-their-health-implications-a-review/Biotech Asia+1ORCID+1

6. J. He, R. Mao, Z. Shao, J. L. Wright, D. A. Kerr, C. J. Boushey, and F. Zhu, "An End-to-End Food Image Analysis System," *arXiv preprint arXiv:2102.00645*, 2021. [Online]. Available: https://arxiv.org/abs/2102.00645Purdue Engineering+1arXiv+1

7. A. Khasgiwala, "Word Recognition in Nutrition Labels with Convolutional Neural Network," M.S. thesis, Dept. of Computer Science, Utah State Univ., Logan, UT, 2018. [Online]. Available: https://digitalcommons.usu.edu/etd/7101/DigitalCommons@USU+1Grafiati+1

8. P. Apoorva, M. Bindushree, and H. A. Bhamati, "Nutrition Facts Label Processing using Image Segmentation and Token Matching based on OCR," *International Journal of Applied Engineering Research*, vol. 11, no. 9, pp. 6591–6597, 2016. [Online]. Available: https://www.researchgate.net/publication/318440108

ABSTRACT

This project demonstrates a system that scans ingredient labels on food products to extract additives like preservatives and food colors. Using OCR for text extraction, the system identifies these additives and provides information on their potential harmful effects and Acceptable Daily Intake (ADI) based on scientific data. The goal is to enhance consumer awareness regarding the safety of food additives.

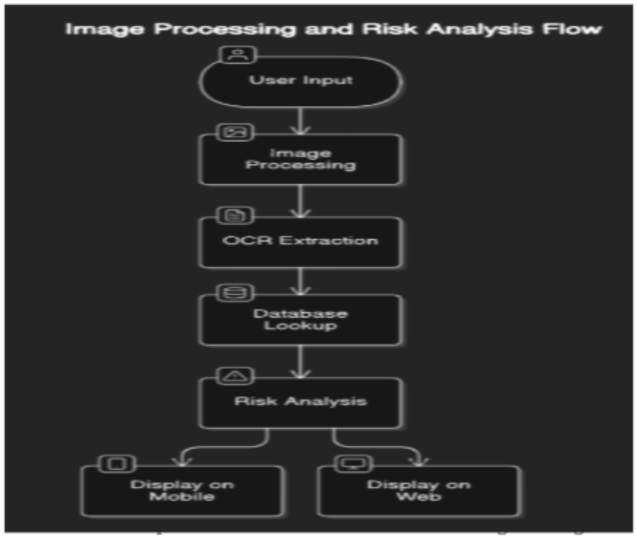
25-04-2025 Department of Information Science & Engineering 3

PROBLEM STATEMENT

Consumers often find it difficult to understand food additives and their health risks due to complex names on labels. This project aims to address this by using OCR technology to scan ingredient labels, identify additives, and provide health risk information along with safe consumption levels (ADI), helping consumers make informed dietary choices.

25-04-2025 Department of Information Science & Engineering 6

SYSTEM DESIGN & METHODOLOGY



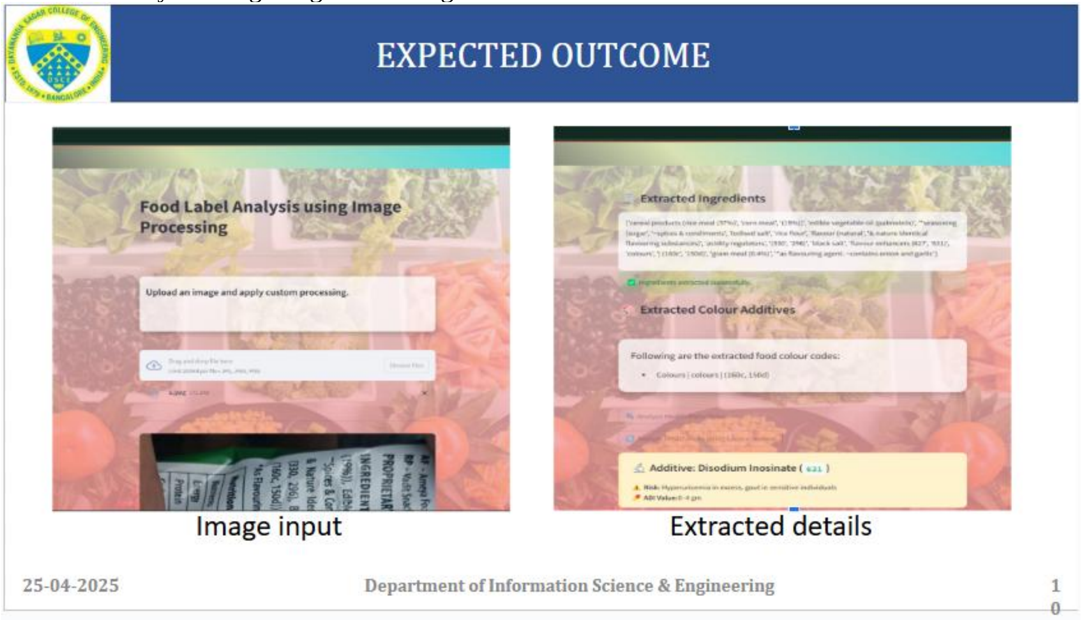
25-04-2025 8

SYSTEM DESIGN & METHODOLOGY

Methodology:

1. **Data Acquisition:** Users upload an image containing the ingredient list of a food product.
2. **Image Processing:** The system processes the image to enhance the quality and make the text more readable for OCR.
3. **Text Recognition:** OCR technology is used to convert the processed image into text.
4. **Text Parsing:** The text is analyzed to find additive names and codes.
5. **Additive Identification:** Extracted additives are matched with a database containing potential side effects and ADI values.
6. **Result Display:** The processed information is presented to the user in a structured format, highlighting harmful additives.

25-04-2025 Department of Information Science & Engineering 9


EXPECTED OUTCOME
Food Label Analysis using Image Processing
Upload an image and apply custom processing.
Extracted Ingredients
Extracted Colour Additives
Following are the extracted food colour codes:
Additive: Disodium Inosinate
Image input
Extracted details
25-04-2025
Department of Information Science & Engineering
10