A PROJECT REPORT

ON

Food Recognition and Nutrition Estimation using CNN Model

Submitted in the partial fulfillment of the requirements for the award of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

SUBMITTEDBY

NAKKA SHIVA	MOHAMMAD SHARIQ ALI	JATROTH SHIVA
21BK1A05C8	21BK1A05B2	21BK1A0580

Under the guidance of
Mr. Mohd Arshad, M. Tech, (Ph. D)
ASSISTANT PROFESSOR DEPARTMENT OF CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

St. Peter's Engineering College (UGC Autonomous)

Approved by AICTE, New Delhi, and NAAC Accredited with 'A' Grade,

Affiliated to JNTU, Hyderabad, Telangana.

2024-2025

St. PETER'S ENGINEERING COLLEGE
UGC - AUTONOMOUS
Affiliated to JNTUH, Approved by AICTE, Accredited by NAAC with "A" Grade, NBA Programe Accredited (EEE, CSE, ECE)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that a Project entitled **"FOOD RECOGNITION AND NUTRITION ESTIMATION USING CNN MODEL"** is carried out **by Nakka Shiva (21BK1A05C8)**, **Mohammad Shariq Ali (21BK1A05B2)**, **Jatroth Shiva (21BK1A0580)**, in partial fulfillment for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work done by her/him under my supervision during the academic year 2024– 2025.

INTERNAL GUIDE
Mr. Mohd Arshad, M. Tech., (Ph. D)
Assistant Professor,
Department of CSE

HEAD OF THE DEPARTMENT
Dr. P. Amarendra Reddy
Professor&HOD
Department of CSE

PROJECT COORDINATOR
Mr. A. Senthil Murugan,M.E.,M.B.A.,(Ph.D)
Assistant Professor
Department of CSE

EXTERNALEXAMINER

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACKNOWLEDGEMENT

We sincerely express our deep sense of gratitude to **Guide Name,** for her valuable guidance, encouragement and cooperation during all phases of the project.

We are greatly indebted to our Project Coordinator **Mr. A. Senthil Murugan,** for providing valuable advice, constructive suggestions and encouragement without whom it would not been possible to complete this project.

It is a great opportunity to render our sincere thanks to **Dr. P. Amarendra Reddy,** Head of the Department, Computer Science and Engineering for his timely guidance and highly interactive attitude which helped us a lot in successful execution of the Project.

We are extremely thankful to our Principal **Dr.Nagu Chandra Sekhar Reddy,** who stood as an inspiration behind this project and heartfelt for his endorsement and valuable suggestions.

We respect and thank our Administrative Director **Mr. T. Anuraag Reddy**, Academic Director **Mrs. T. Saroja Reddy** and secretary **Sri. T. V. Reddy,** for providing us an opportunity to do the project work at **St. PETER'S ENGINEERING COLLEGE** and we are extremely thankful to them for providing such a nice support and guidance which made us to complete the project.

We also acknowledge with a deep sense of reverence, our gratitude towards our parents, who have always supported us morally as well as economically. We also express gratitude to all our friends' who have directly or indirectly helped us to complete this project work. We hope that we can build up on the experience and knowledge that we have gained and make a valuable contribution towards the growth of the society in coming future.

NAKKA SHIVA (21BK1A05C8)

MOHAMMAD SHARIQ ALI (21BK1A05B2)

JATROTH SHIVA (21BK1A0580)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE VISION

To be a renowned Educational Institution that moulds Students into Skilled Professionals fostering Technological Development, Research and Entrepreneurship meeting the societal needs.

INSTITUTE MISSION

IM1: Making students knowledgeable in the field of core and applied areas of Engineering to innovate Technological solutions to the problems in the Society.

IM2: Training the Students to impart the skills in cutting edge technologies, with the help of relevant stake holders.

IM3: Fostering a conducive ambience that inculcates research attitude, identifying promising fields for entrepreneurship with ethical, moral and social responsibilities.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT VISION

To be a global leader in Computer Science Education, Research and Development, Innovation through industry collaboration and cultivating future-ready technologists.

DEPARTMENT MISSION

DM1: Cultivate experiential learning through state-of-the-art technologies and algorithms to address real-world interdisciplinary challenges.

DM2: Collaborate with industry and academia to equip students with cutting-edge skills and prepare them for dynamic careers.

DM3: Strengthen theoretical and applied research in Computer Science & Engineering, fostering innovation, impactful contributions, and instilling ethical values and social responsibility in all endeavors.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Innovate and contribute to research, driving technological advancements in industry and society.

PEO2: Excel as leaders in software development, ensuring industry and government excellence.

PEO3: Lead entrepreneurial ventures, fostering growth in computer science-driven technologies at national and global levels.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

PO1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and teamwork: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. Life-long learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM SPECIFIC OBJECTIVES (PSO'S)

PSO-1: Design and develop computing subsystems for data storage, communication, information processing, and knowledge discovery.

PSO-2: Design algorithms for real-world problems, focusing on execution and complexity analysis while considering security, cost, quality, and privacy parameters in software development.

St. PETER'S ENGINEERING COLLEGE
UGC - AUTONOMOUS
Affiliated to JNTUH, Approved by AICTE, Accredited by NAAC with "A" Grade, NBA Programe Accredited (EEE, CSE, ECE)

DEPARTMENTOFCOMPUTERSCIENCEANDENGINEERING

DECLARATION

We declare that a Major Project entitled **"Food Recognition and Nutrition Estimation using CNN Model"** is an Original Work submitted by the following group members who have actively contributed and submitted in partial fulfillment for the award of degree in **"Bachelor of Technology in Computer Science and Engineering",** at **St. Peter's Engineering College**, Hyderabad, and this project work has not been submitted by me to any other college or university for the award of any kind of degree.

Group No: 19

Program: B.Tech

Branch: CSE

Major Project Title: Food Recognition and Nutrition Estimation using CNN Model

Date Submitted:

Name	Roll Number	Signature
Nakka Shiva	21BK1A05C8	
Mohammad Shariq Ali	21BK1A05B2	
Jatroth Shiva	21BK1A0580	

TABLE OF CONTENTS

ABSTRACT

The "Food Recognition and Nutrition Estimation using CNN Model" project aims to revolutionize culinary exploration and recipe discovery by employing Convolutional Neural Networks (CNNs) in tandem with traditional recommendation methodologies. Unlike conventional recommendation systems that primarily rely on collaborative filtering or content-based approaches, our system integrates CNNs to process both image and text data associated with recipes. This novel approach offers a more holistic understanding of recipe content, enabling more accurate and personalized recommendations tailored to individual preferences and dietary requirements.

The system begins with the collection and preprocessing of a diverse dataset encompassing recipe images and corresponding text descriptions. These descriptions include ingredients, cooking instructions, and other relevant details. The CNN models are then trained on this dataset to extract meaningful features from recipe images and text, leveraging the spatial hierarchies captured by the convolutional layers for image recognition and the semantic understanding encoded by the subsequent layers for text analysis.

The integration of CNNs enhances the recommendation process in several key ways. Firstly, the image recognition capabilities enable the system to identify visually similar dishes, facilitating recommendations based on aesthetic appeal and presentation. Users can explore recipes that resonate with their culinary preferences simply by browsing through visually engaging images. Additionally, the CNNs are trained to recognize key ingredients and cooking techniques from recipe text, allowing the system to generate recommendations based on ingredient availability and dietary restrictions. For instance, users can input their pantry items or dietary preferences, and the system will suggest recipes that align with their needs.

Furthermore, the system incorporates traditional recommendation techniques such as collaborative filtering and content-based filtering to complement the CNN-based approach. By combining multiple recommendation strategies, our system provides a more comprehensive and accurate recipe recommendation experience. Evaluation of the system's performance will be conducted using standard metrics such as accuracy, precision, and recall, ensuring the effectiveness and reliability of the recommendation engine.

Ultimately, the "Food Recognition and Nutrition Estimation" aims to empower users to discover and explore a wide variety of culinary delights tailored to their tastes and dietary preferences. Whether seeking quick and easy weeknight meals or gourmet creations for special occasions, our system will serve as a valuable tool for culinary enthusiasts, novices, and seasoned chefs alike, fostering a deeper appreciation for the art and joy of cooking.

LIST OF FIGURES

1. INTRODUCTION

In the contemporary digital landscape, the abundance of online recipe content presents both opportunities and challenges for culinary enthusiasts. While the internet offers a vast repository of culinary inspiration, navigating through countless recipes to find dishes that align with individual tastes, dietary preferences, and ingredient availability can be daunting. Traditional recommendation systems often fall short in providing personalized and relevant recipe suggestions, lacking the ability to capture the nuanced characteristics of recipes, such as visual appeal and ingredient combinations.

To address these challenges, we introduce the Food Recognition and Nutrition Estimation, a novel platform designed to revolutionize recipe discovery and meal planning. Leveraging state-of-the-art technologies such as Convolutional Neural Networks (CNNs) alongside traditional recommendation methodologies, our system aims to provide users with intuitive, personalized, and visually engaging recipe recommendations. By analyzing both image and text data associated with recipes, the system extracts meaningful features to enhance recommendation accuracy. Through a user-friendly interface, users can input their preferences, explore diverse recipe options, and receive tailored suggestions that cater to their culinary needs.

This introduction sets the stage for the Food Recognition and Nutrition Estimation, highlighting the significance of personalized recipe recommendations in simplifying the meal planning process and inspiring culinary exploration. Throughout this project, we will delve into the development, implementation, and evaluation of the system, aiming to empower users to discover, explore, and enjoy a wide variety of culinary delights tailored to their tastes and preferences.

1.1 MOTIVATION

The Food Recognition and Nutrition Estimation is driven by the need to simplify recipe discovery and meal planning in today's digital age. With an abundance of online recipe content, users often struggle to find dishes that match their preferences, dietary needs, and ingredient availability. Traditional recommendation systems fall short in capturing the nuanced aspects of recipes, such as visual appeal and ingredient combinations. By integrating Convolutional Neural Networks (CNNs) with conventional techniques, our system aims to address these challenges. CNNs analyze both image and text data, allowing for a comprehensive understanding of recipe content. This approach enables the system to recommend visually appealing dishes and recognize key ingredients and cooking techniques. Ultimately, our goal is to empower users to explore new culinary horizons, discover diverse recipes, and simplify their cooking experience in a digital world.

1.2 PROBLEM DEFINITION

In today's digital era, the vast array of online recipe content presents a challenge for users seeking personalized and relevant meal suggestions. Traditional recommendation systems often fail to capture the intricate nuances of recipes, such as visual appeal and ingredient combinations, leading to a disconnect between users' preferences and recommended dishes. Additionally, users frequently encounter difficulties in finding recipes that align with their dietary restrictions and ingredient availability. This highlights the need for an innovative solution that combines advanced technologies with traditional recommendation methodologies to provide tailored recipe recommendations.

The Food Recognition and Nutrition Estimation aims to address these challenges by leveraging Convolutional Neural Networks (CNNs) to analyze both image and text data associated with recipes. By integrating CNNs with conventional recommendation techniques, the system seeks to provide users with personalized and visually engaging recipe suggestions that cater to their tastes, dietary preferences, and ingredient availability. Ultimately, the goal is to simplify the recipe discovery process, inspire culinary exploration, and enhance the overall cooking experience for users in an increasingly digitalized world.

1.3 OBJECTIVE OF THE PROJECT

The objective of the Food Recognition and Nutrition Estimation project is to develop an innovative platform that transforms the way users discover and plan meals. Leveraging Convolutional Neural Networks (CNNs), the system aims to analyze recipe images and text data, extracting meaningful features to enhance recommendation accuracy. By integrating CNNs with traditional recommendation methodologies, such as collaborative and content-based filtering, the system strives to provide personalized recipe suggestions tailored to individual tastes and dietary needs. Emphasis is placed on recognizing visually appealing dishes, identifying key ingredients, and considering cooking techniques to improve recommendation relevance. Through user-friendly interfaces and iterative refinement based on performance metrics and user feedback, the system aims to empower users to explore diverse culinary options effortlessly. Ultimately, the project seeks to simplify meal planning, inspire culinary creativity, and enrich the overall cooking experience for users.

1.4 LIMITATIONS

The scope of the Food Recognition and Nutrition Estimation project encompasses several crucial aspects aimed at developing an efficient and user-centric platform. This includes collecting and preprocessing a diverse dataset of recipe images and text descriptions to ensure data quality. The project involves designing and training Convolutional Neural Network models to extract relevant features from recipe images and text, thereby facilitating accurate recommendations. Integration with traditional recommendation techniques like collaborative filtering and content-based filtering expands the system's capability to provide personalized recipe suggestions. Additionally, user interface design is pivotal, ensuring intuitive interaction for inputting preferences, viewing recommendations, and offering feedback. Evaluation using standard metrics such as accuracy and precision, alongside iterative refinement based on user feedback, ensures the system's continual improvement. Deployment options, either standalone or integration into existing platforms, guarantee accessibility to a broad user base. While the project emphasizes system functionality, it does not extend to content creation or implementing advanced features beyond the scope of CNNs for recipe analysis.

2. LITERATURE SURVEY

[1] Seda Kul & Ahmet Sayar , "A Smart Recipe Recommendation System Based on Image Processing and Deep Learning", Kocaeli University, Baki Komsuoğlu bulvarı No:515, Umuttepe, 41001, Kocaeli, Turkey, 2021.

While technology facilitates our lives, it also allows us to lead a quality, efficient energy and to do more productive work in less time. One of our basic needs that takes our time is to eat. One of the problems people face is, ''What can I cook today?'' It is the question. The Smart Recipe Suggestion System (SRSS) is a system that can make food recommendations to the person with the materials available. This system is designed as a mobile application, including the deep learning method, to answer people's questions during the day. This paper proposes an approach that recognizes the person's materials with image processing technology and presents the most suitable meal suggestions to be made with those materials. Recipes are collected from web sites through RabbitMQ. Food items are captured from images by using object detection processes through deep learning. Available food items are modeled as NoSQL and stored to MongoDB document based databases. For the real-world application Firebase mobile cloud platform is used.

[2] M. Gim, D. Park, M. Spranger, K. Maruyama and J. Kang, "RecipeBowl: A Cooking Recommender for Ingredients and Recipes Using Set Transformer," in IEEE Access, vol. 9, pp. 143623-143633, 2021,

Countless possibilities of recipe combinations challenge us to determine which additional ingredient goes well with others. In this work, we propose RecipeBowl which is a cooking recommendation system that takes a set of ingredients and cooking tags as input and suggests possible ingredient and recipe choices. We formulate a recipe completion task to train RecipeBowl on our constructed dataset where the model predicts a target ingredient previously eliminated from the original recipe. The RecipeBowl consists of a set encoder and a 2-way decoder for prediction. For the set encoder, we utilize the Set Transformer that builds meaningful set representations. Overall, our model builds a set representation of an leave-one-out recipe and maps it to the ingredient and recipe embedding space. Experimental results demonstrate the effectiveness of our approach. Furthermore, analysis on model predictions and interpretations show interesting insights related to cooking knowledge.

[3] Devis Bianchini, Valeria De Antonellis, Nicola De Franceschi, Michele Melchiori, "PREFer: A prescription-based food recommender system" University of Brescia, Department of Information Engineering, via Branze 38, 25123 Brescia, Italy, 2017.

In the literature, several researches on food recommendation and automatic menu generation have been proposed, taking into account different aspects, such as personal and cultural preferences, health and religion constraints, menu composition and recipe co-occurrence. However, recommending recipes and menus, which not only meet the user's preferences, but are also compliant with best food habits, is still an open issue. This paper presents the PREFer food recommender system, apt to provide users with personalized and healthy

menus, taking into account both user's short/long-term preferences and medical prescriptions.

[4] W. Min, S. Jiang and R. Jain, "Food Recommendation: Framework, Existing Solutions, and Challenges," in IEEE Transactions on Multimedia, vol. 22, no. 10, pp. 2659-2671, Oct. 2020.

A growing proportion of the global population is becoming overweight or obese, leading to various diseases (e.g., diabetes, ischemic heart disease and even cancer) due to unhealthy eating patterns, such as increased intake of food with high energy and high fat. Food recommendation is of paramount importance to alleviate this problem. Unfortunately, modern multimedia research has enhanced the performance and experience of multimedia recommendation in many fields such as movies and POI, yet largely lags in the food domain. This article proposes a unified framework for food recommendation, and identifies main issues affecting food recommendation including incorporating various context and domain knowledge, building the personal model, and analyzing unique food characteristics. We then review existing solutions for these issues, and finally elaborate research challenges and future directions in this field. To our knowledge, this is the first survey that targets the study of food recommendation in the multimedia field and offers a collection of research studies and technologies to benefit researchers in this field.

[4] Mayumi Ueda, Mari Takahata, and Shinsuke Nakajima, "User's Food Preference Extraction for Personalized Cooking Recipe Recommendation" Kyoto University Yoshida Nihonmatsu-cho, Sakyo-ku, Kyoto, Kyoto 606–8501, Japan, 2018.

There are many websites and researches that involve cooking recipe recommendation. However, these websites present cooking recipes on the basis of entry date, access frequency, or the recipe's user ratings. They do not reflect the user's personal preferences. We have proposed a personalized recipe recommendation method that is based on the user's food preferences. For extracting the user's food preferences, we use
his/her recipe browsing and cooking history. In this paper, we present a method for extracting the user's preferences. In the experimental results, extracting the user's favorite ingredients were detected with a 60 to 83% of precision. And extracting the unfavorite ingredients were detected with 14.7% of precision, and 58% of recall. Furthermore, the F-measure value for extraction of favorite ingredients was 60.8% when we focused on the top 20 ingredients.s

2.1 EXISTING SYSTEM

Traditional recipe recommendation methods typically rely on collaborative filtering and content-based filtering techniques. Collaborative filtering analyzes user interactions and preferences to generate recommendations based on similarities between users or items. Content-based filtering, on the other hand, recommends items similar to those previously liked by the user, focusing on item attributes such as ingredients, cuisine type, or cooking methods.

2.2 DISADVANTAGES OF EXISTING SYSTEM

Lack of Personalization: Traditional recommendation methods such as collaborative filtering and content-based filtering may fail to provide personalized recommendations tailored to individual user preferences, dietary restrictions, and culinary interests. Users may receive generic suggestions that do not align with their specific tastes and needs.

Limited Consideration of Visual Elements: Many existing methods overlook the visual aspects of recipes, such as presentation and aesthetic appeal. While content-based filtering may consider textual attributes like ingredients and cuisine type, it may not account for the visual appeal of dishes, which can be a crucial factor in recipe selection.

Inability to Capture Complex Relationships: Collaborative filtering techniques rely on historical user interactions and similarities between users or items to generate recommendations. However, they may struggle to capture intricate relationships or preferences, especially in cases where users have diverse tastes or dietary restrictions.

Overlooking Contextual Information: Existing methods may not effectively leverage the rich contextual information available in recipe descriptions, including cooking techniques, cultural significance, or occasions. This can lead to recommendations that lack relevance or fail to consider the broader context in which recipes are used.

Reliance on Manual Feature Extraction: Image-based approaches often rely on manual feature extraction or simplistic machine learning models, which may not capture the complex visual patterns and associations present in recipe images. This can limit the effectiveness of image-based recommendation systems and lead to suboptimal results.

2.3 PROPOSED SYSTEM

The proposed Food Recognition and Nutrition Estimation aims to overcome the limitations of existing methods by integrating advanced technologies such as Convolutional Neural Networks (CNNs) with traditional recommendation methodologies. The system will analyze both image and text data associated with recipes to provide personalized, visually engaging, and contextually relevant recommendations.

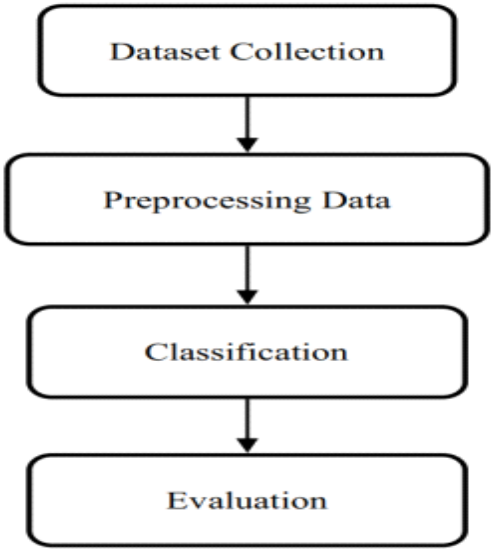
2.4 ADVANTAGES OF PROPOSED SYSTEM

- **Convolutional Neural Networks (CNNs):** The core of the proposed system will be CNN models trained to analyze recipe images and extract meaningful visual features. These CNNs will be capable of recognizing key ingredients, cooking techniques, and

visual similarities among dishes, enhancing the accuracy and relevance of recipe recommendations.

- **Text Analysis:** In addition to image recognition, the system will analyze recipe text descriptions using natural language processing techniques to extract important attributes such as ingredients, cuisine type, and cooking instructions. This text analysis will complement the visual features extracted by the CNNs, providing a more comprehensive understanding of recipe content.
- **Integration with Recommendation Techniques:** The proposed system will integrate CNN-based image and text analysis with traditional recommendation methodologies, such as collaborative filtering and content-based filtering. By combining multiple recommendation strategies, the system will provide more accurate and personalized recipe suggestions tailored to individual user preferences and dietary requirements.
- **User Interaction Interface:** The system will feature an intuitive and user-friendly interface that allows users to input their preferences, dietary restrictions, and ingredient availability. Users will be able to explore recommended recipes, view detailed information about each dish, and provide feedback to further refine future recommendations.
- **Evaluation and Refinement:** The performance of the proposed system will be evaluated using standard metrics such as accuracy, precision, and recall. User feedback and interaction data will be continuously collected and analyzed to refine the recommendation algorithms and improve the overall user experience.

Block Diagram



2.5 FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and business proposal is put forth with

a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical feasibility
- Technical feasibility
- Social feasibility

A. Technical Feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

B. Economical Feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

C. Social Feasibility:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3. ANALYSIS

3.1 REQUIREMENT SPECIFICATION

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional Requirements: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

- Authentication of user whenever he/she logs into the system
- System shutdown in case of a cyber-attack
- A verification email is sent to user whenever he/she register for the first time on some software system.

Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

Examples of non-functional requirements:

- Emails should be sent with a latency of no greater than 12 hours from such an activity.
- The processing of each request should be done within 10 seconds
- The site should load in 3 seconds whenever of simultaneous users are > 10000

3.2 SOFTWARE REQUIREMENTS

Operating System	: Windows 7/8/10
Server side Script	: HTML, CSS, Bootstrap & JS
Programming Language	: Python
Libraries	: Flask, Pandas, Mysql.connector, Os, Smtplib, Numpy

IDE/Workbench	: PyCharm
Technology	: Python 3.6+
Server Deployment	: Xampp Server
Database	: MySQL

3.3 HARDWARE REQUIREMENTS

Processor	- I3/Intel Processor
Hard Disk	- 160GB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA
RAM	- 8GB

4. ARCHITECTURE

4.1 SYSTEM ARCHITECTURE

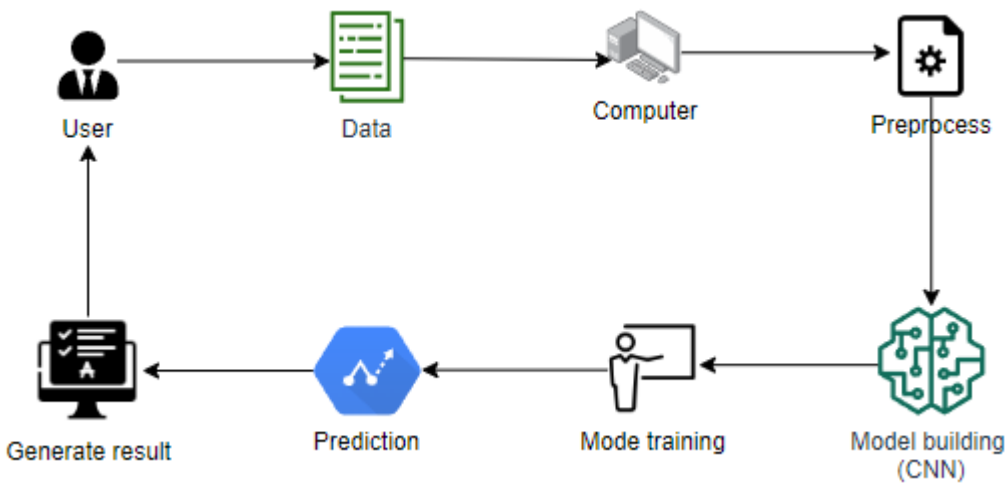


Fig 4.1:System Architecture

1. System:
   1.1 Create Dataset:
   In this module, the dataset containing images for recipie prediction is divided into two subsets - the training dataset and the testing dataset. This split is typically done with a test size of around 20-30%. The training dataset is used to teach the model, while the testing dataset is used to evaluate its performance.

2.Data Pre-Processing:
Data preprocessing is a data mining technique which is used to transform the raw data in a useful and efficient format.

(a). Missing Data:
This situation arises when some data is missing in the data. It can be handled in various ways.

(b). Noisy Data:
Noisy data is a meaningless data that can't be interpreted by machines. It can be generated due to faulty data collection, data entry errors etc.

3. User:
   3.1 Register and Login
   In this application, users are required to register and create their own accounts to access the system's functionalities. Upon registration, users can log in using their credentials to avail themselves of the various features and services provided by the application.
   3.2 prediction:
   - It loads the model that which was created from the training part, choose the images from the system. And then the image is changed into array using img_to_array method.
   - For changing binary values we perform matrix division operation. After converting the binary values we load binary converted data to model for prediction.
   - After predicting we will get recipe name.

4.2 USE CASE DIAGRAM

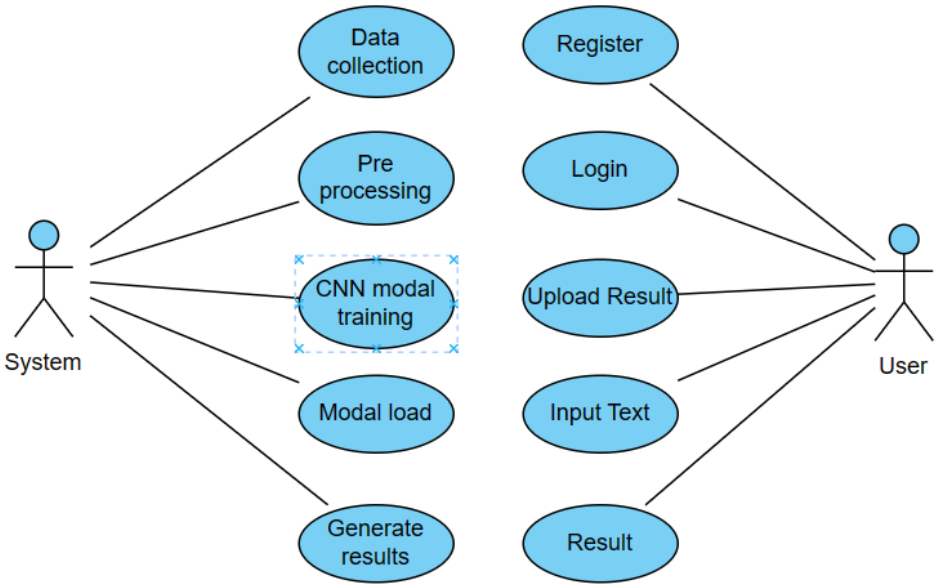


Fig. 4.2 Use Case Diagram for Food Recognition and Nutrition Estimation

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

4.3 CLASS DIAGRAM

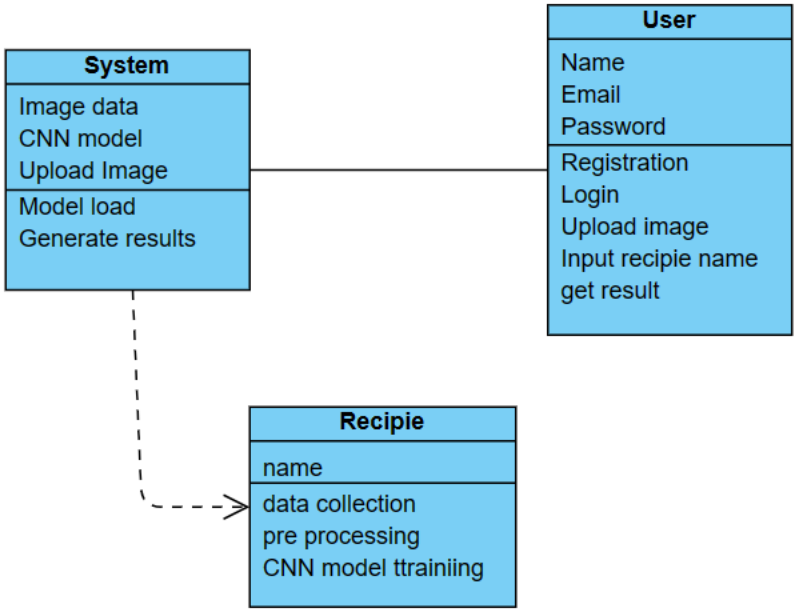


Fig 2. Class Diagram for Food Recognition and Nutrition Estimation

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

4.4 ACTIVITY DIAGRAM

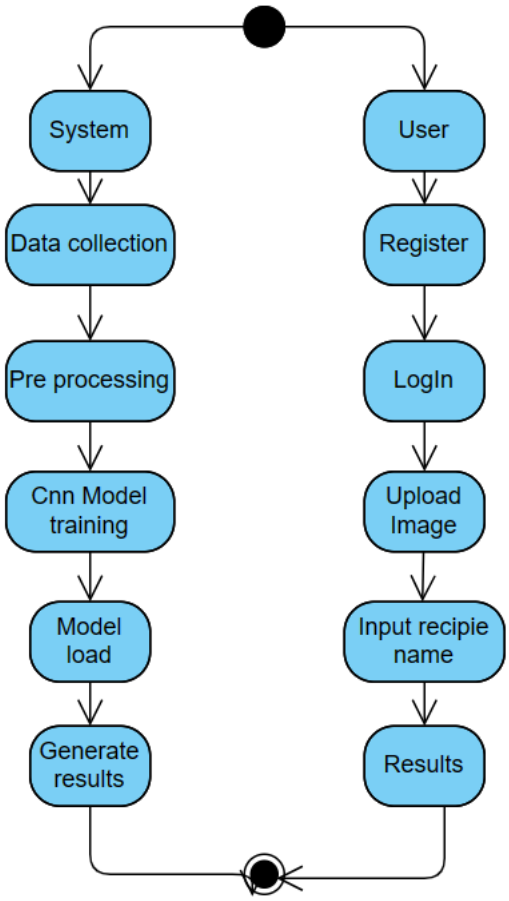


Fig 3. Activity Diagram for Food Recognition and Nutrition Estimation

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

4.5 SEQUENCE DIAGRAM

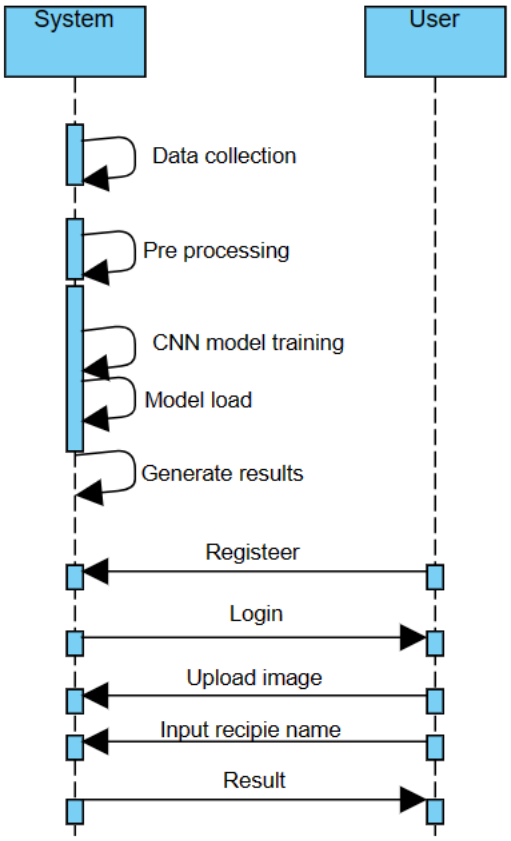


Fig 4. Sequence Diagram for Food Recognition and Nutrition Estimation

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

5. IMPLEMENTATION

5.1 TECHNOLOGIES USED

Convolutional Neural Networks (CNNs) have indeed transformed many areas of computer vision, including the specific domain of food classification. In this context, CNNs are particularly effective because they can directly learn to recognize and differentiate among various types of foods from images, which is crucial for tasks like nutritional analysis, diet tracking, and automated cooking assistance.

In a food classification task, the convolutional layers of a CNN play a critical role. These layers apply numerous filters to the input images to capture and encode different visual features, such as textures of a grilled surface, shapes of different fruits, or colors typical of certain foods. For instance, the early layers might detect edges and colors, while deeper layers could identify more complex patterns like the flakiness of a pastry or the glossiness of fresh vegetables.

Pooling layers further support this process by downsampling the feature maps generated by the convolutional layers, reducing the spatial dimensions of the data. This not only diminishes the computational load and memory usage but also makes the model more robust to variations in the position and size of food items within images. For example, pooling can help the model recognize a pizza whether it's centered in the image or partially cut off at the edge.

Finally, the fully connected layers aggregate all the learned features into a holistic representation, which is used to classify the type of food. In this stage, the CNN might discern whether an image depicts a bowl of pasta or a plate of sushi, based on the learned features.

Furthermore, advances in CNN architectures, such as introducing deeper layers or employing techniques like batch normalization and dropout, have improved their performance and generalization in food classification tasks. Models can be trained on large datasets of food images to achieve high accuracy, making them invaluable for apps and services aimed at food recognition.

Overall, CNNs offer a powerful tool for automated food classification, leveraging their hierarchical learning structure to interpret complex and varied visual data directly from raw pixels, significantly benefiting areas like health monitoring, culinary exploration, and automated food processing.

5.2 SAMPLE CODE

FRONTEND CODE

App.Py

```
from flask import Flask,render_template,flash,redirect,request,send_from_directory,url_for, send_file
import mysql.connector, os
from tensorflow.keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
from keras.preprocessing import image
from tensorflow.keras.models import load_model
import numpy as np

app = Flask(__name__)
app.secret_key = 'your_secret_key_here'

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    port="3306",
    database='foods12'
)

mycursor = mydb.cursor()

def executionquery(query,values):
    mycursor.execute(query,values)
    mydb.commit()
    return

def retrivequery1(query,values):
    mycursor.execute(query,values)
    data = mycursor.fetchall()
    return data

def retrivequery2(query):
    mycursor.execute(query)
    data = mycursor.fetchall()
    return data
```

```
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/register', methods=["GET", "POST"])
def register():
    if request.method == "POST":
        name = request.form['name']
        email = request.form['email']
        password = request.form['password']
        c_password = request.form['c_password']
        if password == c_password:
            query = "SELECT UPPER(email) FROM users"
            email_data = retrivequery2(query)
            email_data_list = []
            for i in email_data:
                email_data_list.append(i[0])
            if email.upper() not in email_data_list:
                query = "INSERT INTO users (name, email, password) VALUES (%s, %s, %s)"
                values = (name, email, password)
                executionquery(query, values)
                return render_template('login.html', message="Successfully Registered! Please go to
login section")
            return render_template('register.html', message="This email ID is already exists!")
        return render_template('register.html', message="Conform password is not match!")
    return render_template('register.html')

@app.route('/login', methods=["GET", "POST"])
def login():
    if request.method == "POST":
        email = request.form['email']
        password = request.form['password']

        query = "SELECT UPPER(email) FROM users"
        email_data = retrivequery2(query)
        email_data_list = []
        for i in email_data:
            email_data_list.append(i[0])

        if email.lower() == "admin@gmail.com":
            if password.lower() == "admin":
```

```
            return redirect("/admin_home")
        else:
            return render_template('login.html', message= "Invalid Password!!")

    if email.upper() in email_data_list:
        query = "SELECT UPPER(password) FROM users WHERE email = %s"
        values = (email,)
        password__data = retrivequery1(query, values)
        if password.upper() == password__data[0][0]:
            global user_email
            user_email = email

            return redirect("/home")
        return render_template('home.html', message= "Invalid Password!!")
    return render_template('login.html', message= "This email ID does not exist!")
return render_template('login.html')

@app.route('/admin_home')
def admin_home():
    return render_template('admin_home.html')

@app.route('/add_recipie', methods = ["GET", "POST"])
def add_recipie():
    if request.method == "POST":
        r_name = request.form['r_name']
        myfile = request.files['image']
        nutrients = request.files['nutrients']
        # ingredients = request.form['ingredients']
        # procedure = request.form['procedure']
        # video_link = request.form['video_link']

        fn = myfile.filename
        mypath = os.path.join('static/images/', fn)
        myfile.save(mypath)

        query = "INSERT INTO recipies(`name`, `image`, `nutrients`) VALUES (%s, %s, %s)"
        values = (r_name, mypath, nutrients)
        executionquery(query, values)
        return render_template('add_recipie.html', message = "Sucessfully added!")
    return render_template('add_recipie.html')

@app.route('/view_recipie', methods = ["GET", "POST"])
```

```
def view_recipie():
    if request.method == "POST":
        id = request.form['id']
        query = "SELECT * FROM recipies WHERE id = %s"
        values = (id, )
        data = retrivequery1(query, values)
        return render_template('view_recipie.html', data = data)

    query = "SELECT * FROM recipies"
    data = retrivequery2(query)
    return render_template('view_recipie.html', datas = data)


@app.route('/update_recipie', methods = ["POST"])
def update_recipie():
    id = request.form['id']
    r_name = request.form['r_name']
    myfile = request.files['image']
    nutrients = request.files['nutrients']
    # ingredients = request.form['ingredients']
    # procedure = request.form['procedure']
    # video_link = request.form['video_link']
    alt = request.form['alt']

    if myfile:
        fn = myfile.filename
        mypath = os.path.join('static/images/', fn)
        myfile.save(mypath)
    else:
        mypath = alt

    query = "SELECT * FROM recipies WHERE id = %s"
    values = (id, )
    data = retrivequery1(query, values)

    query = "UPDATE recipies SET name = %s, image = %s, calaries = %s WHERE id = %s"
    values = (r_name, mypath, nutrients, id)
    executionquery(query, values)

    query = "SELECT * FROM recipies"
    data = retrivequery2(query)
      return render_template('view_recipie.html', datas = data, message = "Successfully
```

```
Updated!")

@app.route('/delete_recipie', methods = ["POST"])
def delete_recipie():
    id = request.form['id']

    query = "DELETE FROM recipies WHERE id = %s"
    values = (id, )
    executionquery(query, values)

    query = "SELECT * FROM recipies"
    data = retrivequery2(query)
      return render_template('view_recipie.html', datas = data, message = "Successfully
Deleted!!")

@app.route('/home')
def home():
    return render_template('home.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/upload', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        myfile = request.files['image']
        fn = myfile.filename
        mypath = os.path.join('static/user_images/', fn)
        myfile.save(mypath)

        classes=["burger","butter_naan","chai","chapati",
                 "chole_bhature","dal_makhani","dhokla","fried_rice",
                 "idli","jalebi","kaathi_rolls","kadai_paneer",
                 "kulfi","masala_dosa","momos","paani_puri", "pakode",
                 "pav_bhaji","pizza","samosa"]

        model=load_model("final_cnn.h5")
        test_img=image.load_img(mypath,target_size=(224,224))
        test_img=image.img_to_array(test_img)
        test_img = np.expand_dims(test_img, axis=0)
        test_img=test_img/255.0
```

```
        # Perform prediction
        prediction = model.predict(test_img)
        result=classes[np.argmax(prediction)]
        print(11111111, result)
        name = result.upper()

        query = "SELECT * FROM recipies WHERE UPPER(name) = %s"
        values = (name, )
        data = retrivequery1(query, values)

        if data:
            prediction = data
        else:
            prediction = "Unknown"

        return render_template('upload.html', path = mypath, prediction = prediction)
    return render_template('upload.html')

@app.route('/search', methods=['GET', 'POST'])
def search():
    if request.method == 'POST':
        text = request.form['text']
        name = text.upper()

        query = "SELECT * FROM recipies WHERE UPPER(name) = %s"
        values = (name, )
        data = retrivequery1(query, values)
        print(11111111111111, data)
        if data:
            prediction = data
        else:
            prediction = "Unknown"

        return render_template('search.html', prediction = prediction)
    return render_template('search.html')

if __name__ == '__main__':
    app.run(debug = True)
```

BACKEND CODE

Model

```
import numpy as np
import pandas as pd
from pathlib import Path
import os.path

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split

import tensorflow as tf

from sklearn.metrics import confusion_matrix, classification_report

from keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
from keras.preprocessing import image
image_dir = Path('Food Classification')
filepaths = list(image_dir.glob(r'**/*.jpg'))
#using glob to target particular image files

labels = list(map(lambda x: os.path.split(os.path.split(x)[0])[1], filepaths))
#separating the class names from the file paths and saving in labels

filepaths = pd.Series(filepaths, name='Filepath').astype(str)
labels = pd.Series(labels, name='Label')

images = pd.concat([filepaths, labels], axis=1)

category_samples = []
for category in images['Label'].unique():
    category_slice = images.query("Label == @category")
    category_samples.append(category_slice.sample(130, random_state=1))
#concatenate category samples
image_df = pd.concat(category_samples, axis=0).sample(frac=1.0,
random_state=1).reset_index(drop=True)
#sample 100% of the data again after shuffling
```

```
image_df['Label'].value_counts()
train_df, test_df = train_test_split(image_df, train_size=0.7, shuffle=True, random_state=1)
#70% training 30% test
#since we are shuffling, random state = 1
#limited memory so we train in batches to recycle memory
train_generator = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input,
    validation_split=0.2
)

test_generator = tf.keras.preprocessing.image.ImageDataGenerator(
    preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input
)
#this train_images will be passed into our fit function
train_images = train_generator.flow_from_dataframe(
    dataframe=train_df,
    x_col='Filepath',
    y_col='Label',
    target_size=(224, 224),
#default image size for mobilenetV2 is 224x224
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
#shuffle after each epoch
    seed=42,
    subset='training'
#only available if validation_split is used to specify whether to use validation subset 0.2 or
training subset

)

validation_images = train_generator.flow_from_dataframe(
    dataframe=train_df,
    x_col='Filepath',
    y_col='Label',
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
```

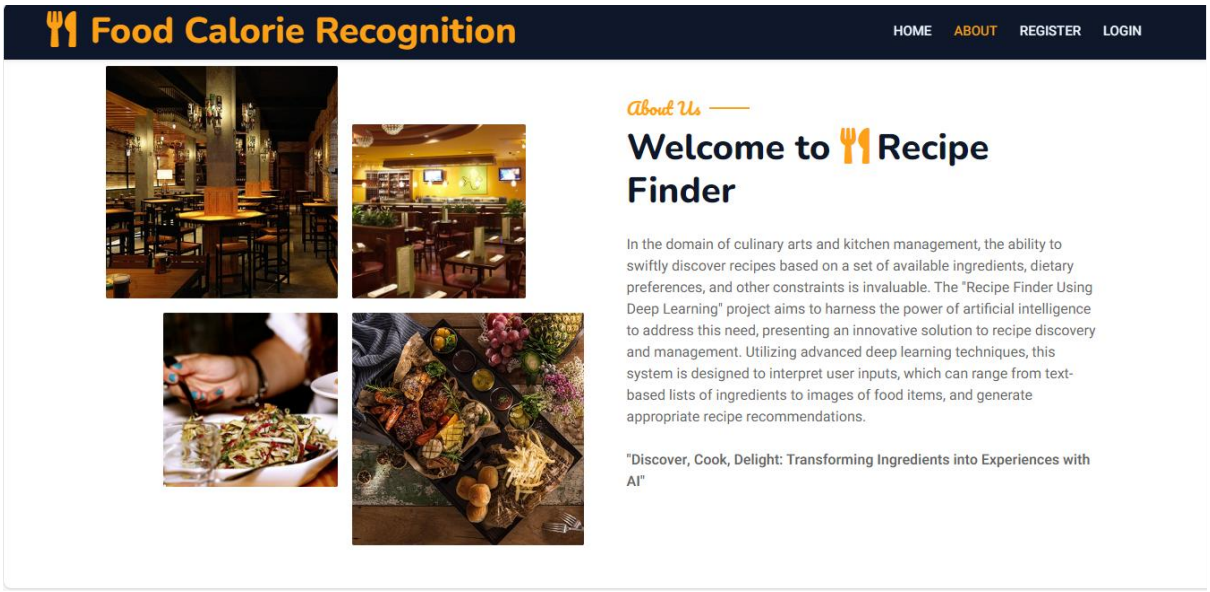
```
    subset='validation'
)

test_images = test_generator.flow_from_dataframe(
    dataframe=test_df,
    x_col='Filepath',
    y_col='Label',
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=False
)
pretrained_model = tf.keras.applications.MobileNetV2(
    input_shape=(224, 224, 3),
    include_top=False,
#we dont wanna keep the classification layer of the og dataset on which the model is pretrained
we just want our dataset's classification layer
#originally trained on imagenet  dataset 1000 classes
    weights='imagenet',
#to keep the same weights
    pooling='avg'
#output is 1d now
)

pretrained_model.trainable = False
#to not change the original imagenet weights
#We are transfering learning of the model so we keep it as it is
#This model is good for feature extraction
#Use the same model, remove the top layer, use your own top layer i.e. classes, dataset
inputs = pretrained_model.input

x = tf.keras.layers.Dense(128, activation='relu')(pretrained_model.output)
#128 neurons
x = tf.keras.layers.Dense(128, activation='relu')(x)

outputs = tf.keras.layers.Dense(20, activation='softmax')(x)
#classification layer
model = tf.keras.Model(inputs, outputs)



print(model.summary())
```

```
model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    #as classes are encoded as vectors by the generator so we use categorical_crossentropy
    metrics=['accuracy']
)

history = model.fit(
    train_images,
    validation_data=validation_images,
    epochs=10,
    #INCREASE NO. OF EPOCHS TO INCREASE ACCURACY (77.46% ACCURACY ACHIEVED ON 100 EPOCHS)
    callbacks=[
        tf.keras.callbacks.EarlyStopping(
            monitor='validation_loss',
            patience=3,
            #when validation loss stops improving for 3 consecutive epochs training will be stopped and best epochs weights are restored
            restore_best_weights=True
        )
    ]
)
results = model.evaluate(test_images, verbose=0)
#gives loss and accuracy for test set
print("Test Accuracy: {:.2f}%".format(results[1] * 100))
model.save("final_cnn.h5")
predictions = np.argmax(model.predict(test_images), axis=1)

cm = confusion_matrix(test_images.labels, predictions)
clr = classification_report(test_images.labels, predictions, target_names=test_images.class_indices, zero_division=0)
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='g', vmin=0, cmap='Blues', cbar=False)
plt.xticks(ticks=np.arange(20) + 0.5, labels=test_images.class_indices, rotation=90)
plt.yticks(ticks=np.arange(20) + 0.5, labels=test_images.class_indices, rotation=0)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
print("Classification Report:\n----------------------\n", clr)
```

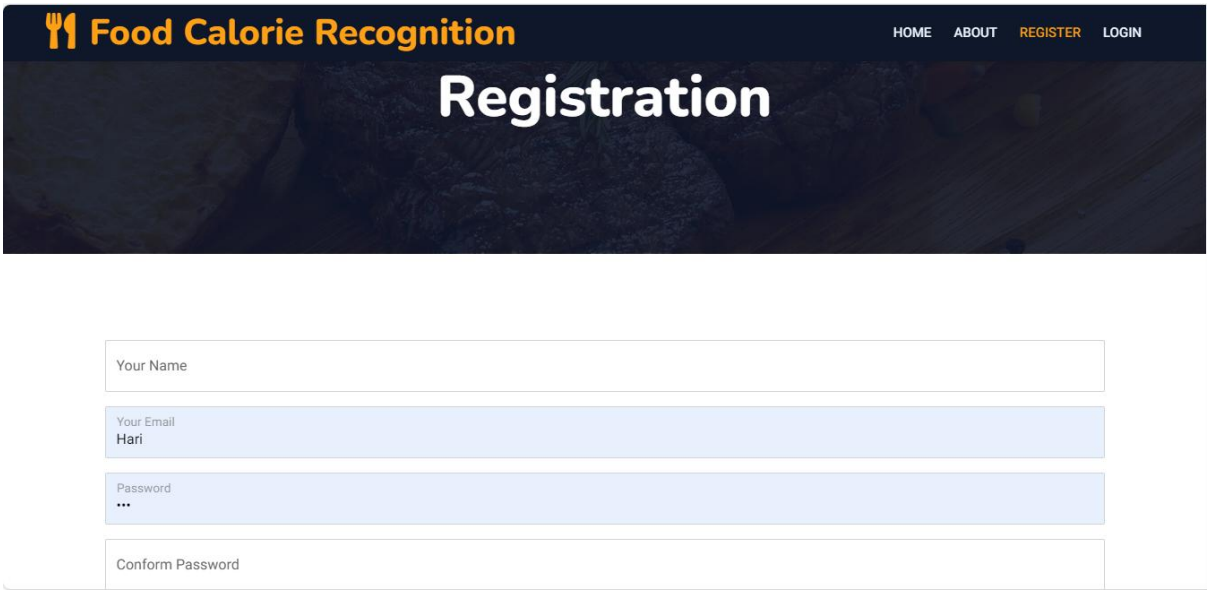
6. SCREENSHOTS

6.1 OUTPUT

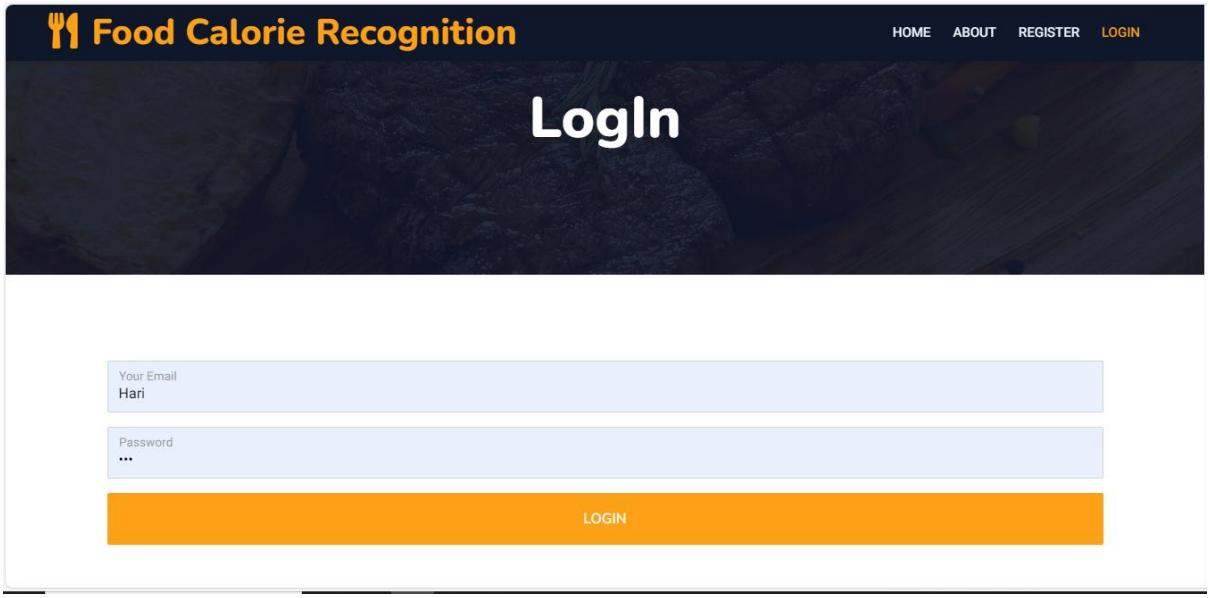
SCREENSHOT 1: OUTPUT – HOME PAGE



SCREENSHOT 2: ABOUT PAGE

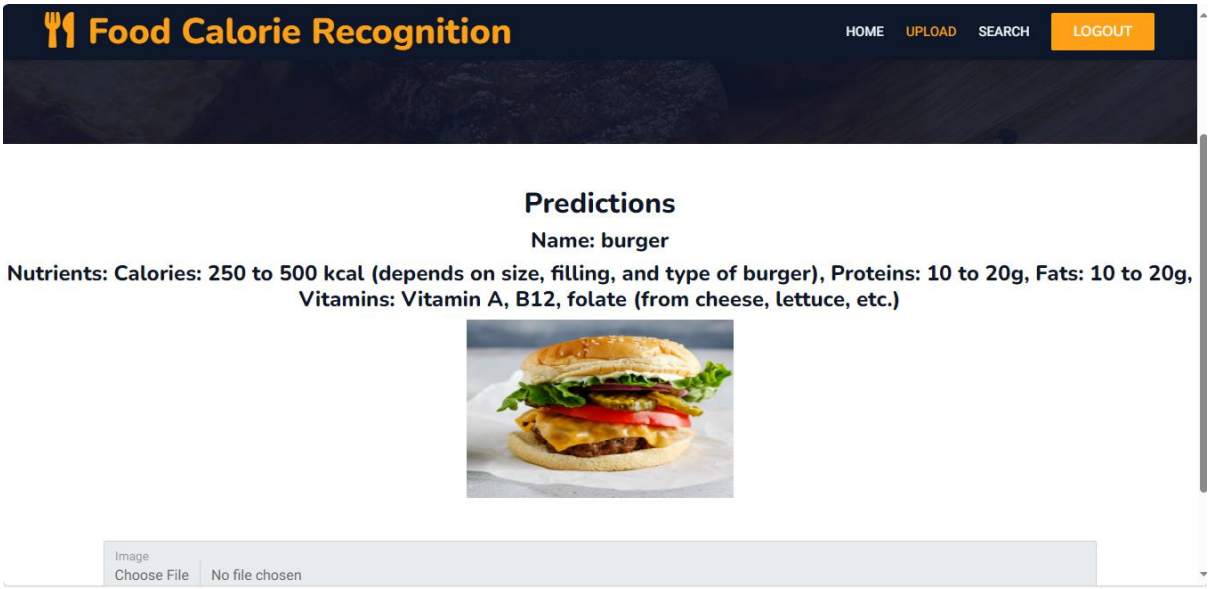


SCREENSHOT 3: REGISTRATION PAGE

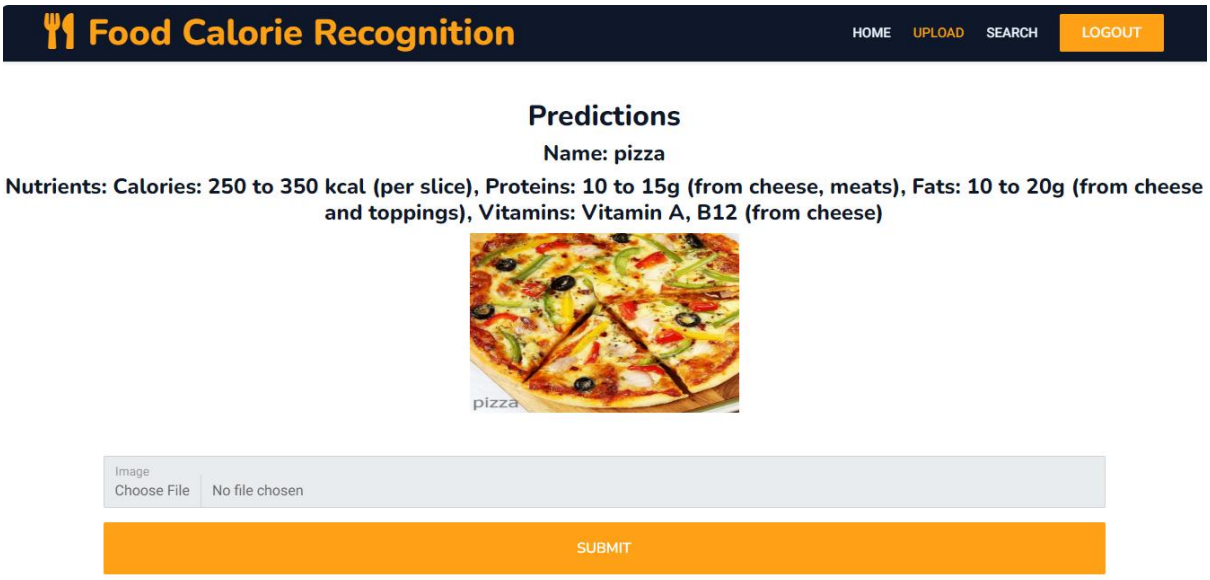


SCREENSHOT 4: LOGIN PAGE

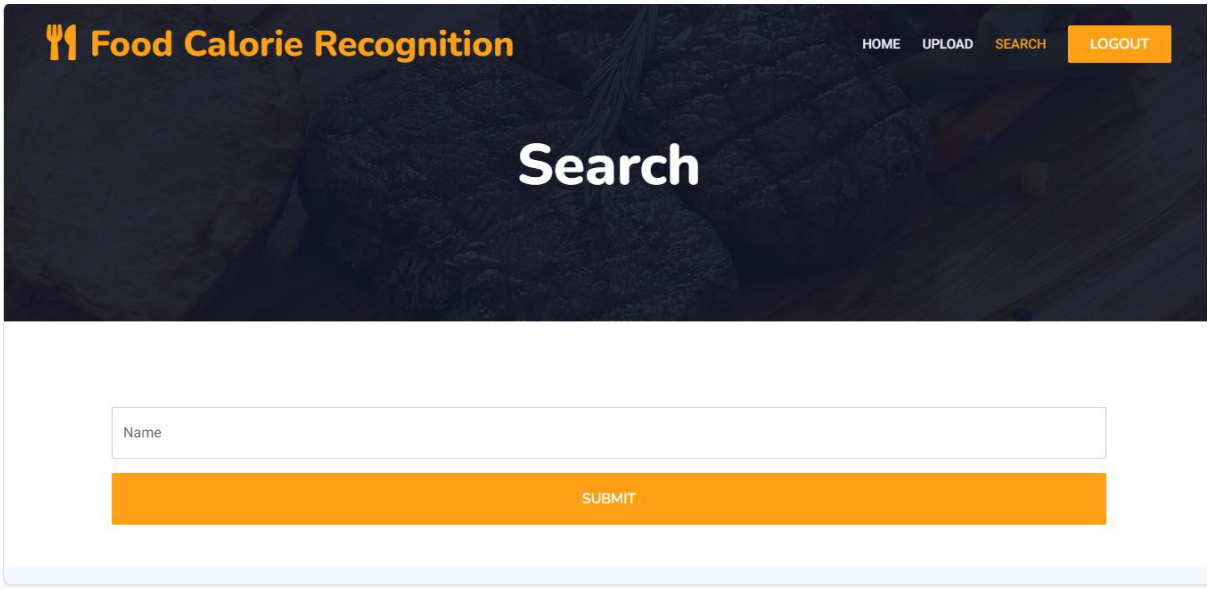
SCREENSHOT 5: USER HOME PAGE



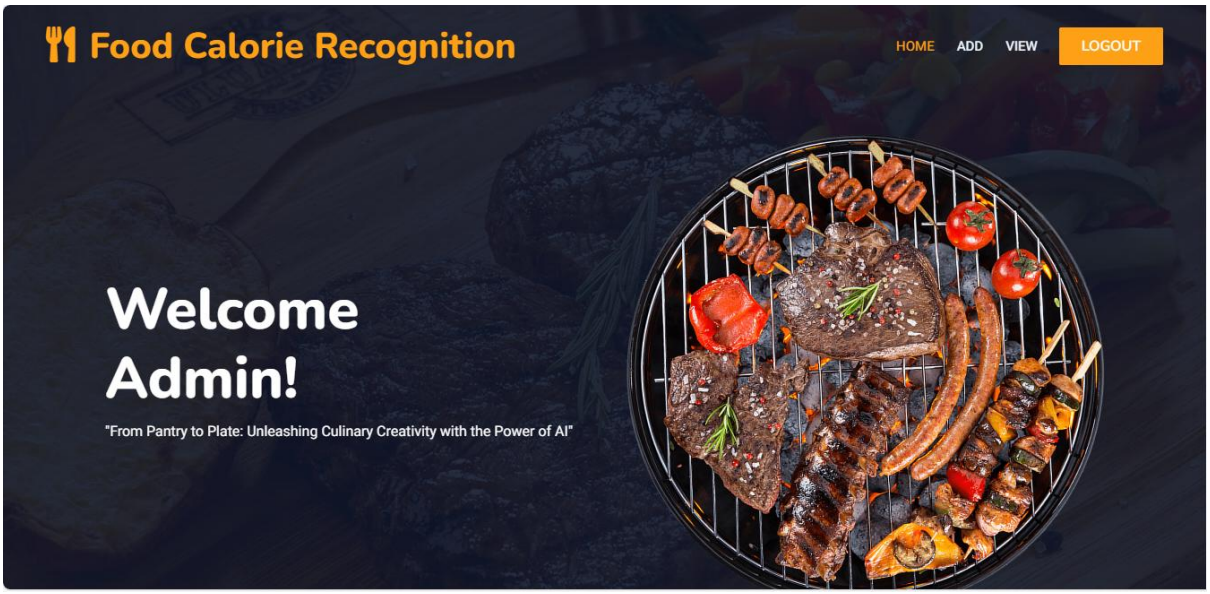
SCREENSHOT 6.1: UPLOAD IMAGE AND RESULT PAGES



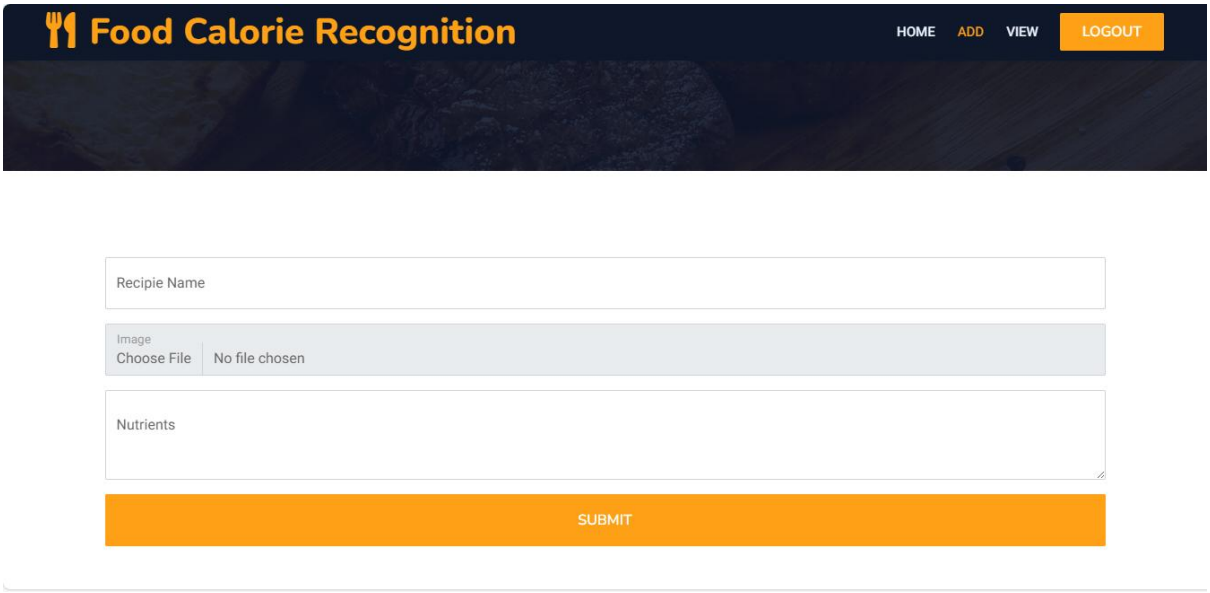
SCREENSHOT 6.2: UPLOAD IMAGE AND RESULT PAGES



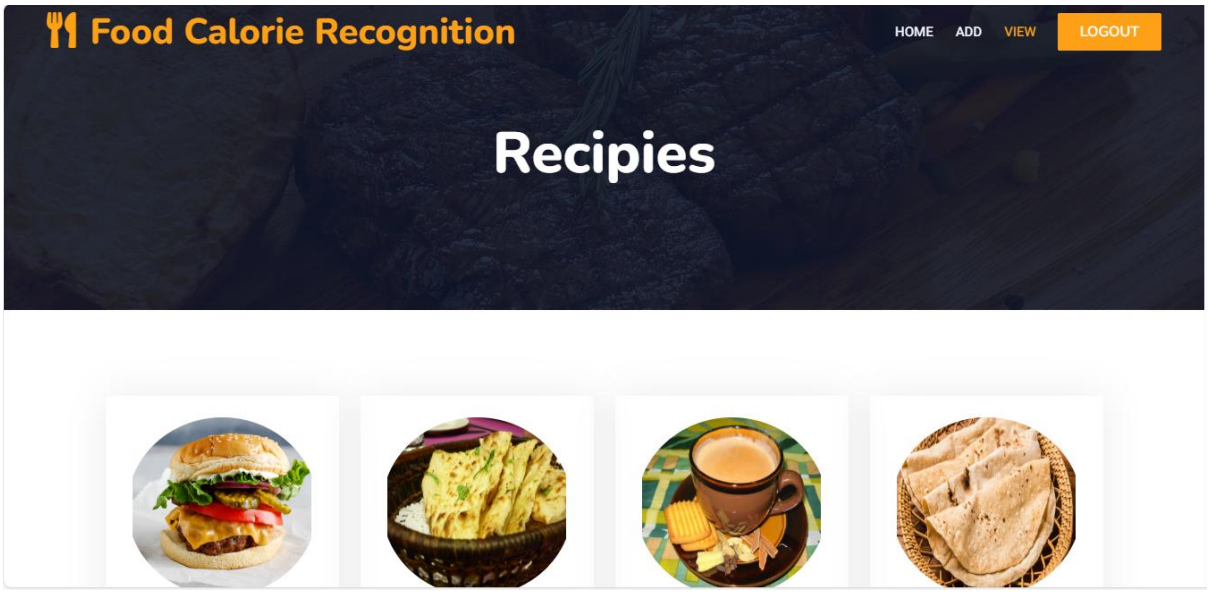
SCREENSHOT 7: TEXT UPLOAD PAGE



SCREENSHOT 8: ADMIN HOME PAGE



SCREENSHOT 9: ADD RECOPIES PAGE



SCREENSHOT 10: VIEW PAGE

7. TESTING

7.1 TYPES OF TESTING

1. UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

2. INGTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

3. ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

4. FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

5. WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6. BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

TEST OBJECTIVES

- All field entries must work properly.
- Pages must be activated from the identified link.

- The entry screen, messages and responses must not be delayed.

FEATURES TO BE TESTED

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.2 TEST CASES

Input	Output	Result
Input image	Recipie detection	Success

7.3 TEST CASES MODEL BUILDING

S.NO	Test cases	I/O	Expected O/T	Actual O/T	P/F
1	Read the dataset.	Dataset path.	Dataset need to read successfully.	Dataset fetched successfully.	P
2	Performing pre-processing on the dataset	Pre-processing part takes place	Pre-processing should be performed on dataset	Pre-processing successfully completed.	P
3	Model Building	Model Building for the clean data	Need to create model using required algorithms	Model Created Successfully.	P
4	Classification	Input image provided.	Output should be sentimental	Model classified successfully	P

8. CONCLUSION & FUTURE SCOPE

8.1 CONCLUSION

In conclusion, the Recipe Recommendation System represents a significant advancement in the field of culinary exploration and meal planning. By integrating Convolutional Neural Networks (CNNs) with traditional recommendation methodologies, the system offers a comprehensive and personalized approach to recipe discovery. Through the analysis of both image and text data associated with recipes, the system provides users with visually engaging, contextually relevant, and highly accurate recipe recommendations tailored to their individual preferences and dietary requirements.

The proposed system addresses the limitations of existing methods by leveraging advanced technologies to capture the complex and nuanced characteristics of recipes. By considering visual elements such as presentation and aesthetic appeal, as well as textual attributes like ingredients and cooking techniques, the system offers a holistic understanding of recipe content, leading to more satisfying and enjoyable culinary experiences for users.
Moving forward, further research and development efforts will focus on refining the recommendation algorithms, enhancing user interaction interfaces, and continuously evaluating the system's performance to ensure its effectiveness and adaptability in meeting the evolving needs of users. Ultimately, the Recipe Recommendation System aims to inspire culinary exploration, foster creativity in the kitchen, and enrich the overall cooking experience for users worldwide.

8.2 FUTURE SCOPE

In future enhancements, the Recipe Recommendation System aims to advance its capabilities in several key areas. Firstly, by refining image recognition through improved CNN models, the system will be better equipped to identify intricate visual patterns within recipe images. Moreover, the integration of multi-modal fusion techniques will allow for a more comprehensive analysis by combining information from both image and text modalities, enhancing recommendation accuracy. User feedback integration will play a crucial role, enabling the system to adapt recommendations dynamically based on user engagement metrics. Additionally, considerations of social and cultural contexts will be integrated, providing more culturally relevant and diverse recipe suggestions. Future developments will also focus on ingredient substitution recommendations, personalized nutritional insights, and integration with smart kitchen appliances for seamless cooking experiences. Community and social features will foster collaboration and knowledge-sharing among users, while ensuring cross-platform compatibility will enhance accessibility across various devices. Lastly, localized recommendations will cater to regional cuisine preferences, ensuring tailored recommendations for users worldwide. These enhancements collectively aim to elevate the Recipe Recommendation System, providing users with more personalized, diverse, and enjoyable culinary experiences.

9. REFERENCES

[1] Seda Kul & Ahmet Sayar , “A Smart Recipe Recommendation System Based on Image Processing and Deep Learning”, Kocaeli University, Baki Komsuoğlu bulvarı No:515, Umuttepe, 41001, Kocaeli, Turkey, 2021

[2] M. Gim, D. Park, M. Spranger, K. Maruyama and J. Kang, "RecipeBowl: A Cooking Recommender for Ingredients and Recipes Using Set Transformer," in IEEE Access, vol. 9, pp. 143623-143633, 2021,

[3] Devis Bianchini, Valeria De Antonellis, Nicola De Franceschi, Michele Melchiori, “PREFer: A prescription-based food recommender system” University of Brescia, Department of Information Engineering, via Branze 38, 25123 Brescia, Italy, 2017.

[4] W. Min, S. Jiang and R. Jain, "Food Recommendation: Framework, Existing Solutions, and Challenges," in IEEE Transactions on Multimedia, vol. 22, no. 10, pp. 2659-2671, Oct. 2020.

[5] Mayumi Ueda, Mari Takahata, and Shinsuke Nakajima, “User’s Food Preference Extraction for Personalized Cooking Recipe Recommendation” Kyoto University Yoshida Nihonmatsu-cho, Sakyo-ku, Kyoto, Kyoto 606–8501, Japan, 2018.