A

MAJOR PROJECT REPORT ON

FOOD AND NUTRITION BASED REVOLUTIONIZING HEALTH AWARENESS USING AI

Submitted in partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted By

K. HARISH GOUD	**218R1A0492**
K. DINESH REDDY	**218R1A0493**
K. KARTHEEK	**218R1A0494**
K. JYOTHIKA	**218R1A0495**

Under the Guidance of

Dr. S. POONGODI
Professor

Department of Electronics and Communication Engineering

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTUH, Accredited by NBA, NAAC)

Kandlakoya(v), Medchal, Telangana.

2024-25

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTUH, Accredited by NBA, NAAC)

Kandlakoya (v), Medchal, Telangana.

Department of Electronics and Communication Engineering

CERTIFICATE

This is to certify that the Major Project work entitled **"FOOD AND NUTRITION BASED REVOLUTIONIZING HEALTH AWARENESS USING AI"** is being submitted by **K. HARISH GOUD** bearing Roll No:**218R1A0492**, **K. DINESH REDDY** bearing Roll No:**218R1A0493**, **K. KARTHEEK** bearing Roll No:**218R1A0494**, **K. JYOTHIKA** bearing Roll No:**218R1A0495** in Batch IV-II semester, Electronics and Communication Engineering is a record Bonafide work carried out by them during the academic year 2024-2025.The results embodied in this report have not been submitted to any other University for the award of any degree.

INTERNAL GUIDE
Dr. S. POONGODI

HEAD OF THE DEPARTMENT
Dr. SUMAN MISHRA

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We sincerely thank the management of our college **CMR ENGINEERING COLLEGE** for providing during our project work.

We derive great pleasure in expressing our sincere gratitude to our principal **Dr.A.S.REDDY** for his timely suggestions, which helped us to complete the project work successfully.

It is the very auspicious moment we would like to express our gratitude to **Dr. SUMAN MISHRA**, Head of the Department, ECE for his consistent encouragement during the progress of this project.

We take it as a privilege to thank our project coordinator **Dr. T. SATYANARAYANA,** Associate professor, Department of ECE for his ideas that led to complete the project work and we also thank his continuous guidance, support and unfailing patience, throughout the course of this work.

We sincerely thank our project internal guide **Dr. S. POONGODI,** Professor, Department of ECE for her guidance and encouragement in carrying out this project.

DECLARATION

We hereby declare that the project work entitled **"FOOD AND NUTRITION BASED REVOLUTIONIZING HEALTH AWARENESS USING AI"** is the work done by us in campus at **CMR ENGINEERING COLLEGE, UGC AUTONOMOUS** Kandlakoya during the academic year 2024-2025 and is submitted as Major Project in partial fulfilment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING** from **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD.**

K. HARISH GOUD	**218R1A0492**
K. DINESH REDDY	**218R1A0493**
K. KARTHEEK	**218R1A0494**
K. JYOTHIKA	**218R1A0495**

CONTENTS

	PAGE NO.
CERTIFICATE	I
DECLARATION BY THE CANDIDATE	II
ACKNOWLEDGEMENT	III
CONTENTS	IV
ABSTRACT	i
LIST OF FIGURES	ii
LIST OF TABLES	iv
LIST OF ABBREVIATIONS	v
CHAPTER-1	
1. INTRODUCTION	6
1.1 PROBLEM STATEMENT	7
CHAPTER-2 LITERATURE SURVEY	9
CHAPTER-3	
3. SYSTEM ANALYSIS	13
3.1 EXISTING SYSTEM	13
3.2 PROPOSED SYSTEM	16
3.3 FUNCTIONAL REQUIREMENTS	22
3.4 NON-FUNCTIONAL REQUIREMENTS	22
3.5 FEASIBILITY ANALYSIS	22
3.5.1 ECONOMICAL FEASIBILITY	23
3.5.2 TECHNICAL FEASIBILITY	23
3.5.3 SOCIAL FEASIBILITY	23
CHAPTER 4	
4. SYSTEM REQUIREMENTS	24
4.1 HARDWARE REQUIREMENTS	24

4.2 SOFTWARE REQUIREMENTS 24

4.3 INSTALL PYTHON STEP-BY-STEP IN WINDOWS AND MAC 25

4.4 INSTALLATION OF PYTHON IN WINDOWS 28

CHAPTER-5

5. SOFTWARE DESIGN 33

5.1. UNIFIED MODELLING LANGUAGE DIAGRAMS 33

5.1.1 USE CASE DIAGRAM 33

5.1.2 ACTIVITY DIAGRAM 34

5.1.3 SEQUENCE DIAGRAM 34

5.1.4 CLASS DIAGRAM 35

CHAPTER-6

6. SYSTEM IMPLEMENTATION 36

6.1 MODULES OF PROPOSED SYSTEM 38

6.1.1 LOAD DATA 38

6.1.2 DATA COLLECTION 39

6.1.3 DATA PRE- PROCESSING 41

6.1.4 FEATURE SELECTION 42

6.1.5 FEATURE EXTRACTION 43

6.1.6 DEEP LEARNING 45

6.1.7 MODEL SELECTION IN DEEP LEARINING 45

6.2 SOFTWARE TECHNOLOGIES 46

6.2.1 GUI DEVELOPMENT IN PYTHON 47

6.2.2 FLASK WEB FRAMEWORK 49

6.3 TESTING 49

CHAPTER-7 RESULTS & DISCUSSION 53

CHAPTER-8
8. CONCLUSION 64
8.1 FUTURE SCOPE 64
CHAPTER-9 REFERENCES 65

ABSTRACT

In recent years, major advances in artificial intelligence (AI) have led to the development of powerful AI systems for use in the field of nutrition in order to enhance personalized dietary recommendations and improve overall health and well-being. However, the lack of guidelines from nutritional experts has raised questions on the accuracy and trustworthiness of the nutritional advice provided by such AI systems. It aims to address this issue by introducing a novel AI-based nutrition recommendation method that leverages the speed and explainability of a deep generative network and the use of novel sophisticated loss functions to align the network with established nutritional guidelines. The use of a variational autoencoder to robustly model the anthropometric measurements and medical condition of users in a descriptive latent space, as well as the use of an optimizer to adjust meal quantities based on users' energy requirements enable the proposed method to generate highly accurate, nutritious and personalized weekly meal plans. Coupled with the ability to provide an unparalleled pool of meals from various cuisines, the proposed method can achieve increased meal variety, accuracy and generalization capabilities, demonstrate the exceptional accuracy of the proposed diet recommendation method in generating weekly meal plans that are appropriate for the users in terms of energy intake and nutritional requirements, as well as the easiness with which it can be integrated into future diet recommendation systems.

LIST OF FIGURES

FIG NO.	FIGURE NAME	PAGE NO.
4.1	Installation of python	26
4.2	Python latest version download	26
4.3	Selecting most recent version	27
4.4	Different version of Python	27
4.5	Open downloaded python	28
4.6	Enabling the path	29
4.7	Setup of Python	29
4.8	Run Command	30
4.9	Testing Python	31
4.10	Windows run command	31
4.11	Saving the file	32
4.12	Execution	32
5.1	Use Case diagram.	34
5.2	Activity diagram flow of control	33
5.3	Message Sequence chart	34
5.4	Class diagram	35
7.1	Home page of the Application	53
7.2	Registration for the application	54
7.3	Sign in using Username and password	54
7.4	Uploading input for the application	55
7.5	Output of pancakes	56
7.6	Output of onion_ rings	56
7.7	Output of pizza	57
7.8	Output of fish_ and_ fries.	57
7.9	Output of chicken_ curry	58
7.10	Output of ice_ cream	58

7.11	Graphical representation of Accuracy Score	61
7.12	Graphical representation of Precision Score	62
7.13	Graphical representation of Recall Score	63
7.14	Graphical representation of F1 Score	64

LIST OF TABLES

TABLE.NO	NAME OF THE TABLE	PAGE NO.
3.1	Comparison result of Existing methods in Accuracy, Precision, Recall and F1-score	11
7.1	Comparison Result of the ML Model for InceptionResNetV2, Xception, NASNetMobile and Extension	45

LIST OF ABBREVIATIONS

Acronym		Abbreviation
CNN	-	Convolutional Neural Networks
CVD	-	Cardiovascular Diseases
DNN	-	Deep Neural Networks
EFSA	-	European Food Safety Authority
GAN	-	Generative Adversarial Networks
GUI	-	Graphical User Interface
LLM	-	Large Language Models
ODR	-	Ontology of Dietary Recommendations
ULM	-	Unified Modeling Language
VAE	-	Variational Autoencoders
WHO	-	World Health Organization

CHAPTER 1

INTRODUCTION

Over the past decade, AI has grown remarkably, giving rise to large deep networks and AI agents with impressive capabilities, even reaching human-level performance, in diverse domains. Such technological breakthroughs have unlocked significant opportunities, but also led to serious risks that include privacy violation, discrimination, as well as the ability of AI systems to achieve their objectives in ways that differs from the intended one [1–3]. In the field of nutrition specifically, AI systems have been widely proposed to provide personalized dietary advice. Nutrition plays a crucial role in adopting and maintaining a healthy lifestyle, while it also prevents the onset of serious non-communicable diseases (NCDs), such as obesity, cardiovascular diseases (CVD) and Type-2 diabetes (T2D)4,5 . In addition, nutritious and balanced meals are regularly incorporated into treatment plans to alleviate the consequences or obstruct the further development of various diseases [6,7] In this regard, AI systems that can automatically recommend personalized dietary meal plans can be immensely beneficial to the well-being of users. However, such AI systems face significant challenges that stem from the complexity of prioritizing actual needs of users 8 . Safety is also a crucial parameter in diet recommendations as unbalanced or harmful diets can lead to malnutrition. Such issues should be properly addressed for AI systems to be universally accepted as trustworthy diet recommenders.

Recently, the introduction of Large Language Models (LLMs) and more specifically of ChatGPT [9,10], has sparked numerous discussions regarding its usage. Leveraging the low complexity, the high speed and an almost infinite pool of meals that it can draw from the web, ChatGPT can be used to make dietary recommendations to users [11]. However, an initial investigation of the safety and credibility of the provided meal recommendations unveiled that ChatGPT can be prone to error [12]. On the other hand, traditional nutrition recommendation systems can achieve increased accuracy as they rely on experts' knowledge and validated nutritional guidelines to provide highly balanced, nutritious and safe meal plans [13–16]. Despite the advantages, such systems suffer from reduced time efficiency and increased complexity due to the use of sophisticated ontologies and rules to filter inappropriate meals. Furthermore, the accuracy of traditional nutrition recommendation systems depends highly on the size and biases of the meal databases used to train them,

limiting their applicability on certain population groups and thus their generalization capabilities.

In an effort to develop accurate and explainable AI-based nutrition recommendation systems, it is necessary to distill experts' knowledge into these systems. To this end, this work proposes a novel AI-based diet recommendation system that can leverage the advantages of AI, such as speed, simplicity and generalization ability with the knowledge acquired from nutritional guidelines (i.e., European Food Safety Authority (EFSA)[17–19] and World Health Organization (WHO)20) to guide the system towards accuracy and robustness. More specifically, the proposed system relies on a deep generative network and sophisticated loss functions to generate highly accurate personalized weekly meal plans in terms of energy intake and nutritional content through the modelling of user-specific information and the alignment of the network with well-defined nutritional rules, respectively. Moreover, leveraging on the ability of LLMs to produce equivalent meals, the proposed method significantly expands its meal database for improved accuracy and generalization ability. The contributions of this work are the following:

- A new deep generative network architecture is proposed to create weekly meal plans by employing sophisticated loss functions to align the network with well-founded nutritional guidelines.
- A novel approach for personalized nutrition recommendation that leverages the ability of LLMs to create an almost infinite pool of meals is proposed.
- The proposed AI-based diet recommendation system is validated on a large group of 3000 virtual user profiles and 1000 real profiles with 91000 daily meal plans, generated using meals from the Protein NAP database [21] (large open-source collection of international meals), showcasing its advantages in terms of explainability and accuracy.

1.1 PROBLEM STATEMENT

Proper nutrition is essential for maintaining a strong immune system and overall well-being. However, many individuals suffer from poor immunity due to inadequate or imbalanced diets, often lacking essential nutrients. This project is motivated by the need to address poor immunity by providing accurate calorie intake and nutrition analysis. By offering detailed insights into users' dietary habits, this AI-powered system empowers

individuals to make informed nutritional choices that support immune health. Unlike meal recommendation systems, this approach ensures that users receive objective, science-backed information without dictating specific meals, allowing them to customize their diets based on their unique needs. Ultimately, this project contributes to the broader goal of improving public health by promoting better nutritional awareness and immune resilience.

CHAPTER 2
LITERATURE SURVEY

In recent years, several works have been focused on the development of healthy food recommendations. In general, recommendation systems can be categorized as content-based (CB), collaborative filtering-based (CF), knowledge-based and deep-learning approaches. In order to provide individualized meal suggestions, content based nutrition recommendation systems analyse the properties and composition of food products as well as the dietary requirements and preferences of users. Harvey et al. aimed to address inadequate nutrition by investigating participant perception of crucial aspects in recipes [22]. They created recommendation algorithms that take into account user preferences for ingredients, combinations and nutritional value. Teng et al. utilized ingredient networks (complement and substitute), which analyse nutritional data and web recipe collections to forecast recipe ratings [23]. This method reveals the connections between ingredients, consumer preferences and basic cooking principles. Gutiérrez et al. proposed an augmented-reality assistant that reads food products barcodes and provides recommendations, focusing on food quality and user preferences [24]. Shandilya et al. proposed a content-based recommender system that gives precedence to users' recent preferences over previous ones, thus adapting to shifting user needs [13]. Their work focused on suggesting meals based on user preferences and dietary advice, ensuring that the daily nutrient needs are satisfied.

Collaborative filtering nutrition recommendation systems leverage user interactions and preferences to provide personalized dietary suggestions. These methods group individuals who have similar eating patterns and make meal recommendations based on user similarity. Ge et al. introduced a mobile app with a food recommendation system based on matrix factorization [25]. Their method models user preferences from user ratings and tags in order to offer personalized recipe suggestions. A different method for personalized nutrition recommendation was proposed by Yuan et al. [26]. This method initially employs k-means clustering to categorize foods into subsets prior to utilizing user-based collaborative filtering to suggest foods aligned with the user's preferences and nutritional balance, considering standard recipes. Toledo et al. developed an approach for daily meal planning that incorporates both nutritional and preference factors [14]. It uses a sorting algorithm to filter the initial meal selection depending on user characteristics.

Following that, an optimization phase creates a meal plan that prioritizes fewer recently consumed foods while aligning it with user preferences and nutritional requirements. On the other hand, user-based and item-based algorithms were tested for their ability to generate top-N suggestions by placing food items based on projected ratings [27]. The experimental results showed that the user-based algorithm outperforms the item-based one in terms of accuracy and coverage. Finally, Rostami et al. proposed a two-phase food recommendation system, which starts by using time-aware collaborative filtering to suggest food items based on previous user consumption and then it forecasts food item ratings using nutritional information [15]. To group comparable people and food products, the system uses a clustering method, which improves recommendation accuracy.

Knowledge-based nutrition recommender systems use specialized knowledge of nutrition and dietary recommendations to give consumers personalized nutritional advice. These systems combine expert knowledge in addition to simple data-driven methods to provide highly individualized meal plans. A personalized knowledge based dietary recommendation method targeting obese users was proposed by Jung and Chung [28]. Their method uses collaborative filtering along with knowledge-based context data to create individualized diet menus. By using context-aware modeling, it also addresses the problem of sparse data and enables mobile users to access tailored menus and recipes for the treatment of obesity. Similarly, Mckensy-Sambola et al. adopted a knowledge-based diet recommendation system that employs user data to deduce the best diets and then provides a list of recipes that satisfy the user's nutritional needs [29]. The Ontology of Dietary Recommendations (ODR) is used to get the most important components in the domain of food recommendations, such as diets, recipes, ingredients, anthropometric indices and food allergies. Another research work employed a collaborative recipe knowledge graph (CRKG) that incorporates user interactions and food-related data to develop a novel food recommendation model [30]. The health-aware food recommendation model utilizes a healthiness matching score and a knowledge aware attention graph convolutional neural network to combine user preferences and health requirements and generate healthy personalized recommendations. Recently, a knowledge-based recommendation framework for precise diet planning across various user groups, including both healthy individuals and those with health conditions, was introduced16. The framework comprises two key layers: a qualitative layer employing expert-derived rules and an ontology [31] for ingredient validation and a quantitative layer utilizing optimization techniques to create daily meal

plans based on specific nutrient requirements. Extensive experiments demonstrated the system's effectiveness in generating appropriate, diverse and tailored meal plans.

Nowadays, deep learning methods have achieved outstanding performance in several research areas including nutrition recommendation. Mokdara et al. proposed a food recommendation system that integrates deep neural networks (DNNs) with user-provided favourite ingredients [32]. Ten, a temporal prediction model forecasts future dish recommendations based on the user's profile and eating habits. Another work proposed an explainable food recommendation system that is based on deep image clustering to incorporate visual content to improve nutrition recommendations [33]. Moreover, a similarity score is adopted to propose foods that match with the user preferences. Iwendi et al. proposed to utilize several deep learning techniques for diet recommendation based on patients' characteristics, such as disease, age, weight and gender [34]. The method was tested on medical datasets from hospitals and it was found that LSTM was the best technique, since it modelled better the medical history. Finally, Zhang et al. proposed a method for sequence-based recommendations using dynamic user-item interactions [35]. More specifically, a Long Short-Term Memory (LSTM) network was employed to capture sequential information and used collaborative filtering to suggest personalized meal plans. However, most existing nutrition recommendation systems lack personalization and provide more generic recommendations that may not be suitable for everyone.

To overcome the limitations of current nutrition recommendation systems, several studies leveraged the advancements achieved by LLMs in several fields for personalized nutrition recommendation. Niszczota et al. [12] used ChatGPT to create diets and accommodate specific food allergies. The authors found out that ChatGPT could produce balanced diets but there were inaccuracies in the proposed diets with respect to food calories and meal portions. Tsai et al. [36] used ChatGPT to provide tailored pregnancy nutrition advice for underserved populations. This method aimed to address health disparities and adverse outcomes linked to low socioeconomic status and inadequate nutrition during pregnancy. Other studies used ChatGPT diet recommendations for diabetes and obesity treatments and stressed out the generic nature of the proposed meals, the limited understanding of context and privacy and security concerns associated with the use of LLMs [11,37].

This work proposes a deep generative network to create accurate weekly meal plans by leveraging both the speed, simplicity and generalization ability of AI and the nutritional experts' knowledge in the form of guidelines. Through the use of novel loss functions that implement the nutritional guidelines and the ability of ChatGPT to create a large meal database, the proposed AI-based diet recommendation system can provide accurate nutritional advice aligned with expert validated rules.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing systems for AI-based dietary recommendations predominantly utilize deep generative networks like Variational Autoencoders (VAE) to create personalized meal plans. These systems model user profiles in latent spaces based on anthropometric and medical data, enabling tailored nutrition advice. Traditional rule-based systems rely on ontologies and nutritionist-defined guidelines to ensure meal accuracy and safety but suffer from high complexity, low scalability, and time inefficiencies. On the other hand, Large Language Models (LLMs) like ChatGPT are employed for dietary recommendations due to their access to a vast array of meals. However, these systems are prone to inaccuracies in caloric and nutrient content, offering generic recommendations that lack proper personalization. Overall, the current systems fail to balance precision, diversity, and scalability effectively.

The current AI-based nutrition systems predominantly focus on meal recommendations rather than precise calorie and nutrient analysis. These systems can be categorized into three main approaches: generative network-based systems, rule-based systems, and large language model (LLM)-based systems. Each of these approaches has its strengths and weaknesses, particularly in terms of accuracy, scalability, and personalization.

Generative network-based systems, such as Variational Autoencoders (VAE) and Generative Adversarial Networks (GANs), use deep learning techniques to model user profiles and generate personalized meal plans. These systems rely on large training datasets to create a variety of meal options, but they often suffer from repetitive meal generation and inaccuracies in nutrient estimation. While they offer high personalization, their accuracy in tracking precise calorie intake and micronutrients is limited, making them less reliable for users who need precise nutritional guidance.

Rule-based nutrition systems, on the other hand, follow predefined guidelines and ontologies developed by dietitians and health organizations. These systems ensure strict adherence to nutritional standards set by bodies like the World Health Organization (WHO)

and the European Food Safety Authority (EFSA). However, rule-based approaches suffer from high complexity, limited scalability, and a lack of adaptability to new dietary trends. Since they rely on fixed food databases, they cannot generate new meal combinations beyond their existing dataset, reducing flexibility.

Large Language Models (LLMs) such as ChatGPT and Bard leverage vast amounts of online data to generate diverse meal plans based on user queries. While LLMs can provide quick and scalable meal recommendations, they lack precision in calorie and nutrient estimations because they do not directly rely on structured nutritional databases. Additionally, their suggestions are often generic and not personalized to individual health needs, leading to potential inaccuracies that could misguide users in managing their dietary intake.

Overall, these existing methods struggle to achieve a balance between accuracy, flexibility, and scalability. While generative models focus on personalization, they may lack accuracy in nutrient tracking. Rule-based systems ensure high accuracy but are difficult to scale and modify, whereas LLMs provide fast recommendations but often fail in nutritional precision. Given these limitations, there is a strong need for an AI-based system that does not focus on meal recommendations but instead provides accurate calorie intake and nutrient analysis to help users make informed dietary decisions, especially in addressing health concerns like poor immunity

The table compares the performance of existing AI-based nutrition systems. Rule-based ontologies have the highest accuracy, while VAE offers moderate performance, and LLM-based systems show the lowest precision due to generic recommendations

Table 3.1. Comparison result of Existing methods in Accuracy, Precision, Recall and F1-score.

System Type	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Variational Autoencoder (VAE)	78.5	74.3	72.8	73.5
Rule-Based Ontologies	85.2	82.1	80.5	81.3
LLM-Based System	70.8	67.4	65.9	66.6

- **Limitations of the Existing System**

Despite advancements in AI-based nutrition systems, the existing methods have several limitations that impact their effectiveness in providing accurate and personalized dietary analysis:

1. Limited Accuracy in Nutrient Estimation – AI models, especially LLM-based and generative systems, often miscalculate calories and nutrient values, leading to unreliable dietary recommendations.

2. Lack of Personalization – Many systems generate generic meal plans that do not adequately consider an individual's specific nutritional needs, dietary restrictions, or health conditions.

3. Repetitive Meal Generation – Variational Autoencoder (VAE)-based systems struggle with diversity in meal plans, often producing repetitive suggestions that limit dietary variety.

4. Scalability Issues in Rule-Based Systems – Traditional rule-based systems rely on predefined ontologies, making them difficult to scale and adapt to new food databases or evolving dietary research.

5. Dependency on Fixed Databases – Rule-based and generative AI models are restricted by the size and biases of their training datasets, which can limit their applicability to different populations.

6. Potential for Incorrect Recommendations – LLM-based systems, which pull data from various sources, may generate misleading or incorrect nutritional guidance, as they lack validation from expert-approved guidelines.

7. Lack of Immune Health Consideration – Existing systems primarily focus on meal planning rather than detailed nutrient analysis, neglecting the role of proper nutrition in improving immunity and overall well-being.

- **Disadvantages of Machine Learning System**

Generative Network Limitations:

- Struggles with meal diversity and may generate repetitive plans.
- Can lack personalization without sophisticated fine-tuning.

Rule-based Systems:

- High time complexity and difficult to scale.
- Dependence on pre-existing meal databases limits adaptability.

LLM-based Systems:

- Inaccuracies in calorie and nutrient estimation.
- Vulnerable to producing generic, non-personalized recommendations.

3.2 PROPOSED SYSTEM

To overcome these limitations, the proposed system integrates advanced deep learning architectures like Convolutional Neural Networks (CNN), Inception Net, and Dense Net to revolutionize dietary recommendation systems. CNNs are employed for efficient feature extraction, accurately modeling complex relationships in user data such as medical history and dietary needs. Inception Net enhances the system's ability to analyse multi-scale features, accommodating diverse user patterns, while Dense Net improves the learning process by leveraging dense connections, ensuring scalability and efficiency. This combination ensures robust user profiling, better clustering, and highly personalized meal recommendations. Additionally, the system integrates ChatGPT to expand meal diversity, but potential inaccuracies are addressed through validation mechanisms implemented in the CNN framework. Sophisticated loss functions ensure adherence to guidelines from organizations like EFSA and WHO, while an optimizer module eliminates caloric mismatches, ensuring precise meal portions. This approach not only achieves high accuracy but also provides better meal variety, user satisfaction, and real-time performance, making it a significant improvement over existing system.

CNNs are employed for feature extraction, enabling the system to recognize different food types, portion sizes, and ingredient compositions with high precision. Inception Net

plays a crucial role in this process by capturing multi-scale features, allowing the system to analyse food images at different levels of detail. This ensures that even complex meals with multiple ingredients are accurately identified. By leveraging multiple convolutional kernels, Inception Net enhances the system's ability to process diverse food types, accommodating variations in portion sizes, ingredient textures, and food presentations.

Dense Net improves the learning process and scalability of the system by ensuring that each layer of the neural network has direct access to all previous layers' information. This results in a more efficient learning process, preventing the loss of critical features during training. Dense Net enables better nutrient estimation and calorie calculation by ensuring stronger feature propagation and improved gradient flow, leading to higher accuracy in identifying nutritional content from food images.

To further enhance accuracy, the system incorporates sophisticated loss functions aligned with nutritional guidelines from organizations like the World Health Organization (WHO) and the European Food Safety Authority (EFSA). These loss functions penalize inaccuracies in calorie and macronutrient estimations, ensuring that the nutritional analysis remains scientifically valid and reliable. Additionally, an optimizer module dynamically refines nutrient calculations, correcting discrepancies caused by image variations and portion estimation errors.

This software project is developed using HTML for the front end and Python for the backend, ensuring a user-friendly interface and efficient processing of food image data. By focusing on precise nutrition analysis rather than meal planning, the proposed system provides users with accurate insights into their dietary intake, helping them monitor calorie consumption, improve immunity, and maintain balanced nutrition. This innovative approach bridges the gap between AI-powered food analysis and real-world dietary needs, offering superior accuracy, personalization, and efficiency compared to existing methods.

- **Step 1**: **Understanding Collections of Dataset**

Since your model processes food images, your dataset likely contains:

- Input: Images of different food items.

- Labels: Nutritional information such as calories, macronutrients (proteins, carbs, fats), or food categories.

Dataset Organization

Your dataset may be structured in one of the following ways:

1. Categorization: Images are classified into food categories (e.g., "Pizza", "Burger", "Salad").

2. Regression: The model predicts caloric values and nutrient breakdowns for each image.

3. Multi-label Classification: The model predicts multiple attributes (e.g., food type + calorie content).

- **Step 2: Preprocessing the Data**

  To train an image classification model effectively, preprocessing the dataset is crucial for improving accuracy and efficiency. One of the primary steps is image rescaling, where pixel values are normalized from a 0-255 range to 0-1 by dividing each pixel by 255. This ensures that the model trains faster and prevents numerical instability. Additionally, data augmentation techniques such as random rotation, width and height shifting, zooming, and horizontal flipping are applied to artificially expand the dataset and introduce variations. These transformations help the model generalize better by making it robust to changes in orientation, scale, and positioning, thus reducing the risk of overfitting.

  To efficiently load and process large datasets, an Image Data Generator is used, which dynamically loads images in batches instead of storing them all in memory. The dataset is split into 80% training and 20% validation, ensuring that the model learns from a diverse set of images while being tested on unseen data. The images are resized to 224x224 pixels to match the input requirements of deep learning models, and a batch size of 8 is used for efficient training. The categorical class mode ensures multi-class labels are properly formatted for classification. These preprocessing steps optimize model performance, enhance generalization, and streamline the training process for improved accuracy.

These are the following requirements to process the dataset

- Rescaling ensures pixel values are between 0 and 1.
- Augmentation (rotation, zoom, flip) prevents overfitting by artificially expanding the dataset.
- Splitting data into training (80%) and validation (20%).
- train_generator and valid_generator will feed images in batches to the model.

- **Step 3: Defining Your Model**

For food recognition tasks, pretrained CNN models like MobileNetV2 are highly effective in extracting meaningful visual features. MobileNetV2, a lightweight and efficient model trained on the ImageNet dataset, serves as the backbone for our food classification model. By freezing its layers, we retain its general feature extraction capabilities while avoiding unnecessary retraining. This ensures that the model leverages previously learned patterns from millions of images, reducing training time and improving accuracy.

To fine-tune the model for food classification, we add custom dense layers on top of the pretrained base. A GlobalAveragePooling2D layer transforms the extracted feature maps into a single feature vector, followed by fully connected layers with 512 and 256 neurons using ReLU activation to capture food-specific details. The final layer consists of 10 neurons with softmax activation, enabling multi-class classification for 10 different food categories. The model is compiled with the Adam optimizer and categorical cross-entropy loss, ensuring efficient learning. This transfer learning approach allows the model to specialize in food classification while benefiting from the robust feature extraction of MobileNetV2.

- **Step 4: Defining Callbacks**

To enhance training efficiency and prevent overfitting, we use callbacks that help monitor and optimize the training process. Early Stopping is implemented to stop training automatically if the validation loss stops improving for a specified number of epochs, preventing unnecessary computation and reducing overfitting. We set

patience=5, meaning training will stop if no improvement is observed for five consecutive epochs, and restore_best_weights =True, ensuring that the model retains the best-performing weights.

Additionally, Model Checkpoint is used to save the best model during training. By setting save_best_only=True, the model is saved only when an improvement in validation performance is detected, preventing overwriting with suboptimal weights. These callbacks are passed to the training process, allowing for automatic monitoring and optimization, ensuring the model achieves the best possible performance without excessive training.

- **Step 5: Training the Model**

Now, we can finally train the model using. fit()!

python

CopyEdit

```
hist1 = model1.fit(
    train_generator,
    epochs=30,
    validation_data=valid_generator,
    callbacks=callbacks)
```

Breaking it Down

1. train_generator: Supplies training data dynamically in batches.
2. epochs=30: Trains for **30 full passes** over the dataset.
3. validation_data=valid_generator: Evaluates performance after each epoch.
4. callbacks: Monitors loss and saves the best model.

- **Step 6: Evaluating Performance**

After training the model, we analyze its performance using the hist1 object, which stores the training history, including loss and accuracy metrics over epochs. To evaluate the model's behavior, we plot the loss curves for both training and validation sets. A significant divergence between the two—where validation loss increases while training loss decreases—indicates overfitting, meaning the model memorizes the training data but struggles with unseen data.

Similarly, accuracy curves help assess the model's classification improvement over epochs. A steady increase in both training and validation accuracy suggests effective learning, whereas a widening gap may indicate overfitting. Using Matplotlib, we visualize these trends, plotting loss against epochs to observe training stability and accuracy curves to track classification improvements. This performance analysis helps determine if additional fine-tuning, such as regularization or data augmentation, is needed to improve generalization.

- **Step 7: Making Predictions**

Once the model has been successfully trained, it can be used to predict food items from new images. This involves preprocessing the input image, passing it through the trained model, and retrieving the predicted food category. The predict food function automates this process by first loading the image using TensorFlow's image module. Since the model expects inputs of a specific size, the image is resized to 224x224 pixels, matching the dimensions used during training. The image is then converted into a numerical array and normalized by dividing pixel values by 255.0, ensuring consistency with the training dataset.

Deep learning models typically require input data in batch format, even for a single image, so we use np.expand_dims() to add an extra dimension to the image array. The processed image is then fed into the trained model (model1), which generates a probability distribution over all possible food categories. The np.argmax() function is used to extract the index of the highest probability class, which corresponds to the model's best guess. Finally, the function returns the predicted food category from a predefined list of class labels. This allows real-time food classification, making the

model practical for applications such as automated dietary tracking, restaurant menu identification, and nutrition analysis.

3.3 FUNCTIONAL REQUIREMENTS

The Functional requirements for a system describe the functionality or the services that the system is expected to provide. These are the statements of services the system should provide and how the system should react to particular inputs and how the system should behave in particular situation. User Registration: User Register with their Registration details. User Login: User Login their account using password Live Inputs: Inputs Given by the User requirement. Load Model: Trained or Tested Model will be load. Predict Output: Output will be predict based on parameters.

3.4 NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements describe the system constraints. Performance: The application should have better accuracy and should provide prediction in less time. Scalability: The system must have the potential to be enlarged to accommodate the growth. Capability: The capability of the storage should be high so the large amount of data can be stored in order to train the model.

3.5 FEASIBILITY ANALYSIS

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 are involved in the feasibility analysis are:

➢ ECONOMICAL FEASIBILITY

➢ TECHNICAL FEASIBILITY

➢ SOCIAL FEASIBILITY

3.5.1 ECONOMICAL FEASIBILITY

This system 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.

3.5.2 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.

3.5.3 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.

CHAPTER-4

SYSTEM REQUIREMENTS

To ensure smooth operation of the food image processing and nutrition analysis system, it is essential to have the appropriate hardware and software configurations. These requirements ensure efficient image processing, deep learning model execution, and seamless user interaction.

4.1 HARDWARE REQUIREMENTS

The following requirements are essential for ensuring the smooth operation of the food image processing and nutrition analysis system.

➢ **Processor** - Pentium –IV

➢ **RAM** - 8 GB (min)

➢ **Hard Disk** - 512 GB

➢ **Key Board** - Standard Windows Keyboard

➢ **Mouse** - Two or Three Button Mouse

➢ **Monitor** - SVGA

4.2 SOFTWARE REQUIREMENTS

The following requirements are essential for ensuring the smooth operation of the food image processing and nutrition analysis system.

❖ **Operating system** - Windows 7 Ultimate.

❖ **Coding Language** - Python.

❖ **Front-End** - Python.

❖ **Back-End** - Django-ORM

❖ **Designing** - Html, CSS, java script.

❖ **Data Base** - MySQL (WAMP Server).

4.3 INSTALL PYTHON STEP-BY-STEP IN WINDOWS AND MAC

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version.

My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheat sheet here. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

- **Step 1:** Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org.

Now, check for the latest and the correct version for your operating system.

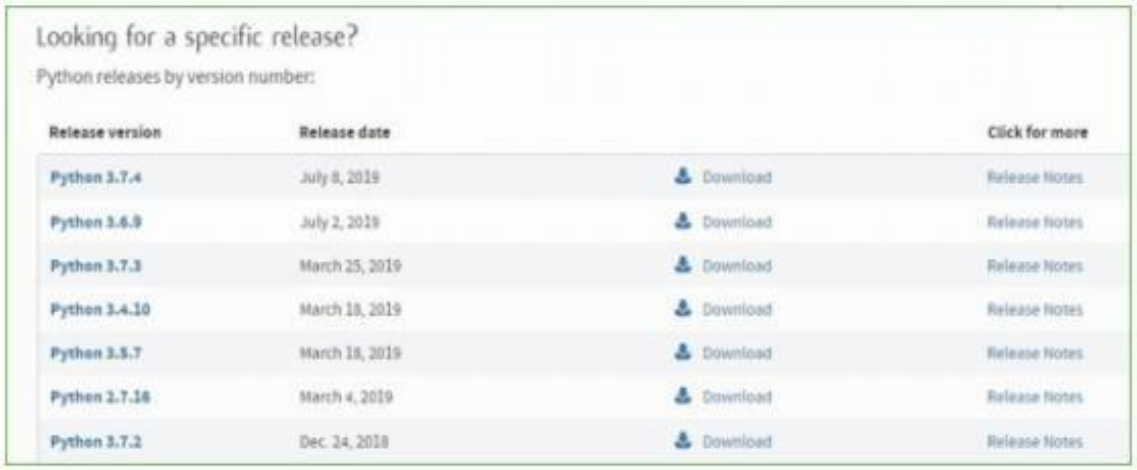
Fig 4.1. Installation of Python

- **Step 2:** Click on the Download Tab

Fig 4.2. Python latest version download

- **Step 3:** You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version.

Here, we are downloading the most recent python version for windows 3.7.4



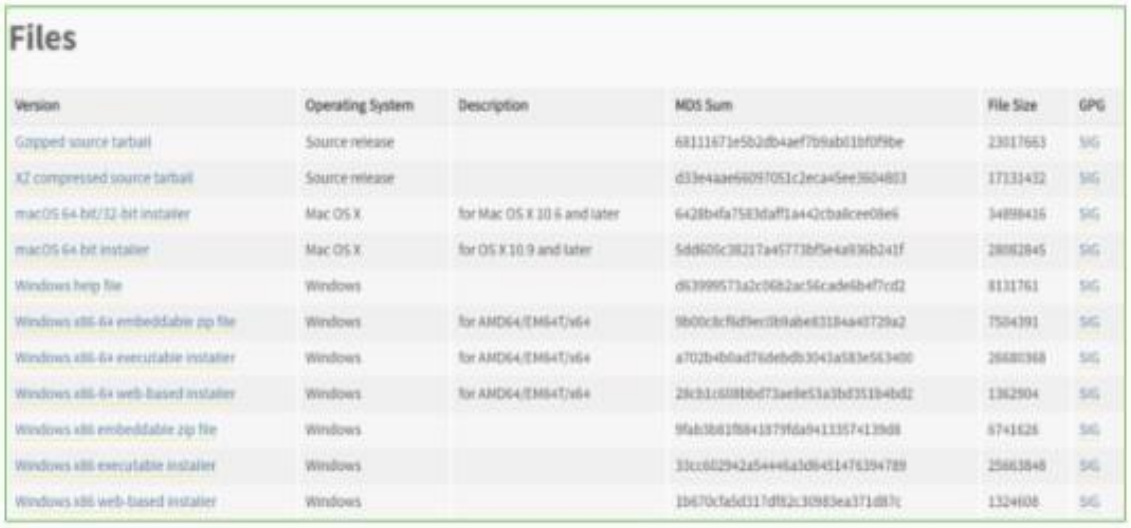
Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Fig 4.3. Selecting most recent version

- **Step 4**: Scroll down the page until you find the Files option.
- **Step 5:** Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa75f3daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a457738f5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999571a2c0682ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00cbcf6df9ec0b9abe63184a403729a2	7584391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76deb0b3041a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c6088bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b61f8841879fda9413357413988	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a1d8451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

Fig 4.4. Different version of python

To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

4.3 INSTALLATION OF PYTHON IN WINDOWS

- **Step 1:** Go to Download and Open the downloaded python version to carry out the installation process.

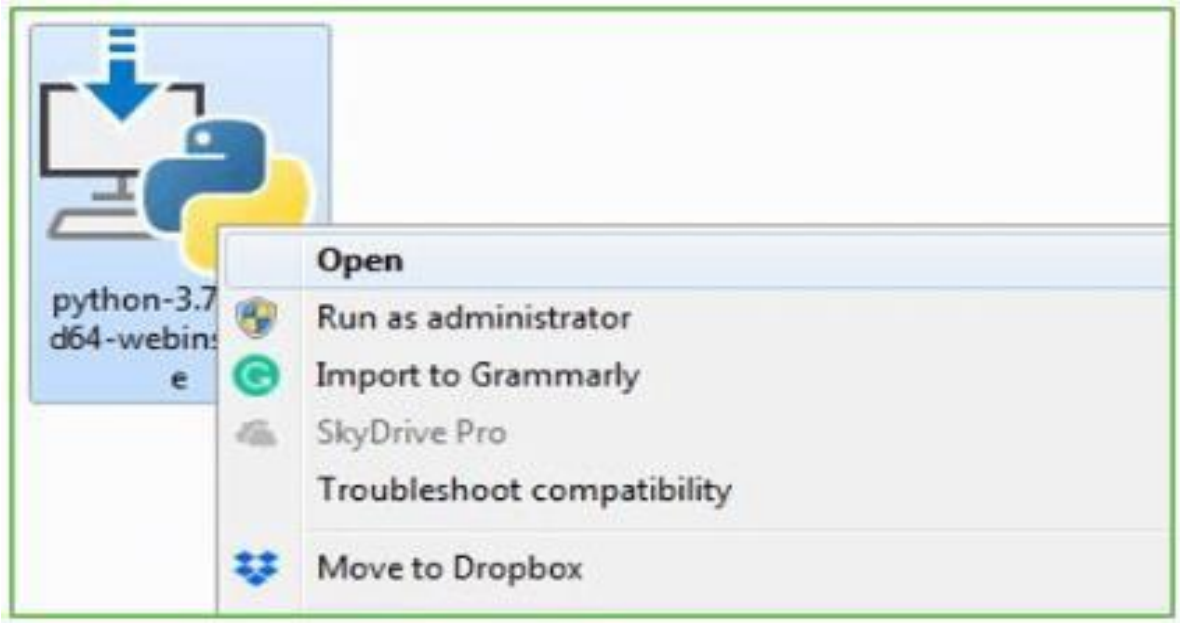


Fig 4.5. Open downloaded python

- **Step 2:** Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.

Fig 4.6. Enabling the path

- **Step 3:** Click on Install NOW After the installation is successful. Click on Close.

Fig 4.7. Setup of Python

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of

minutes. Verify the Python Installation

Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.

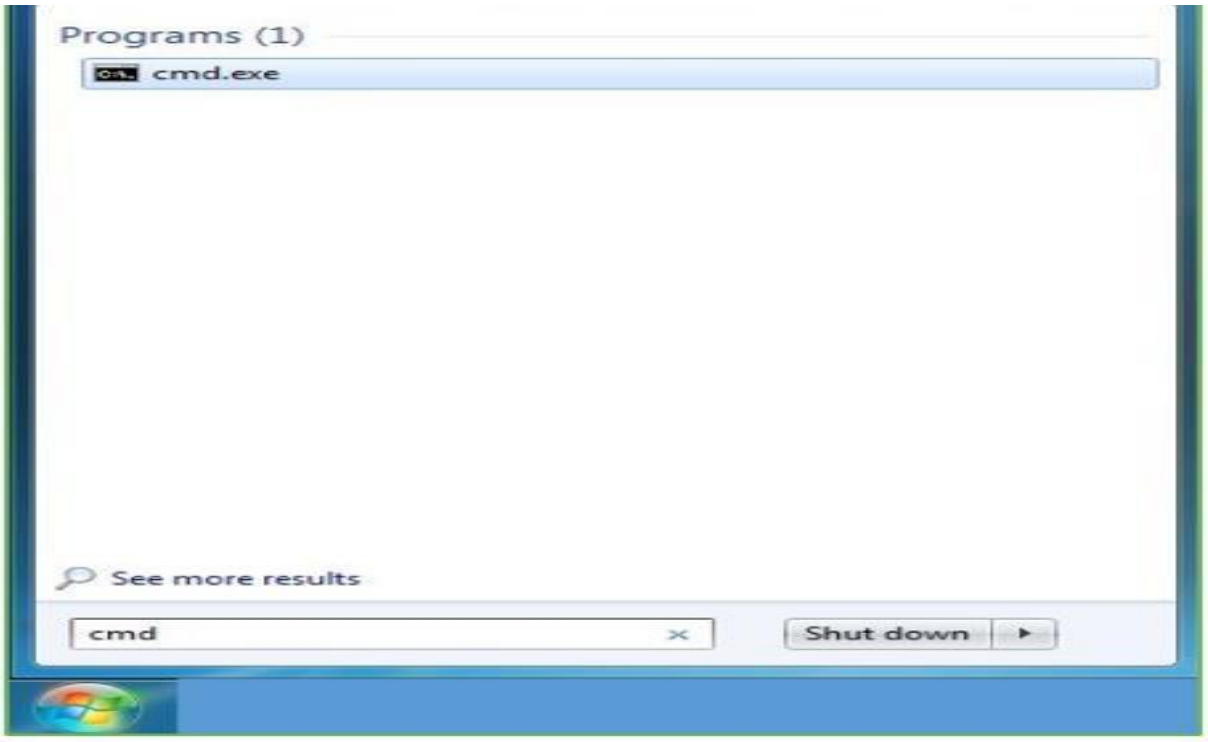
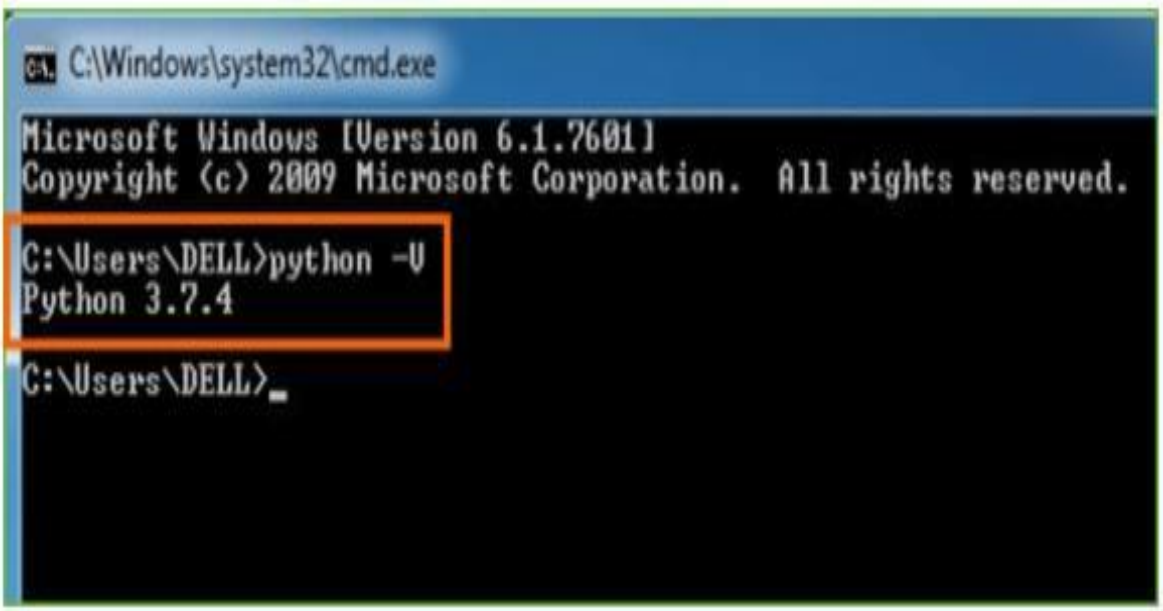


Fig 4.8. Run Command

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.

Note: If you have any of the earlier versions of Python already installed. You must first



uninstall the earlier version and then install the new one.

Fig 4.9. Testing Python

Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE

works **Step 1:** Click on Start

Step 2: In the Windows Run command, type “python idle”.

Fig 4.10. Windows run command

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save

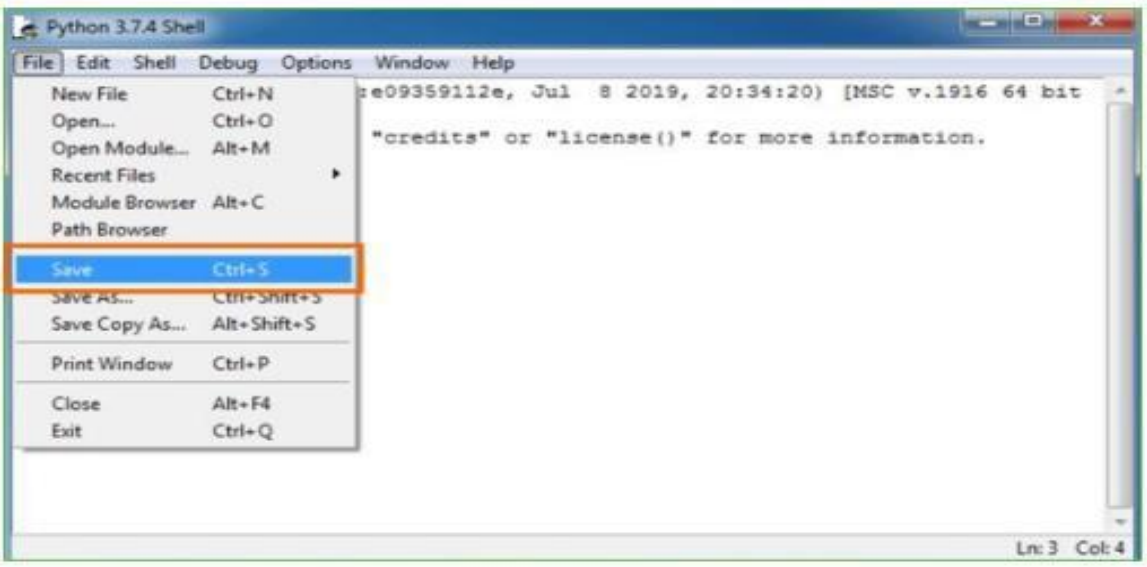


Fig 4.11. Saving the file

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print ("Hey World") and Press Enter.

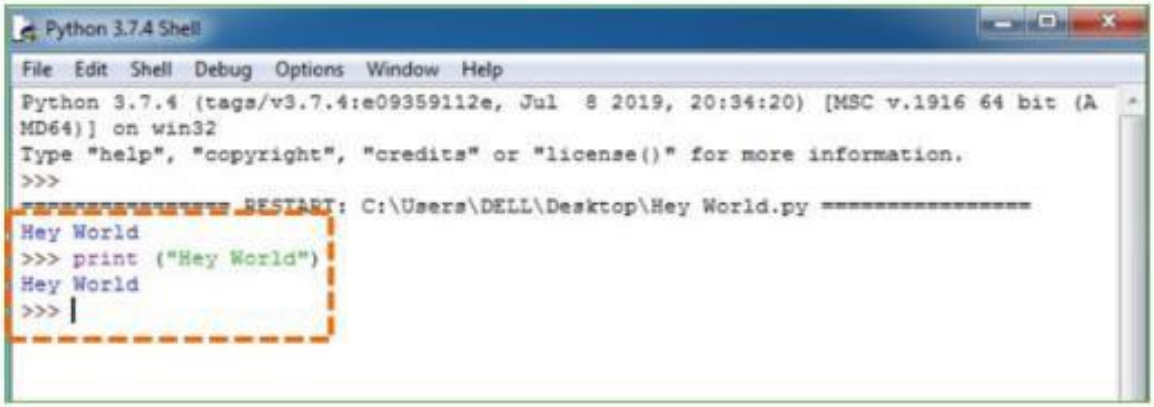


Fig 4.12 Execution

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won't work.

CHAPTER-5

SOFTWARE DESIGN

5.1 UNIFIED MODELLING LANGUAGE DIAGRAMS

UML is a method for describing the system architecture in detail using the blue print. UML represents a collection of best engineering practice that has proven successful in the modeling of large and complex systems. The UML is very important parts of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the helps UML helps project teams communicate explore potential designs and validate the architectural design of the software.

5.1.1 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioural 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.

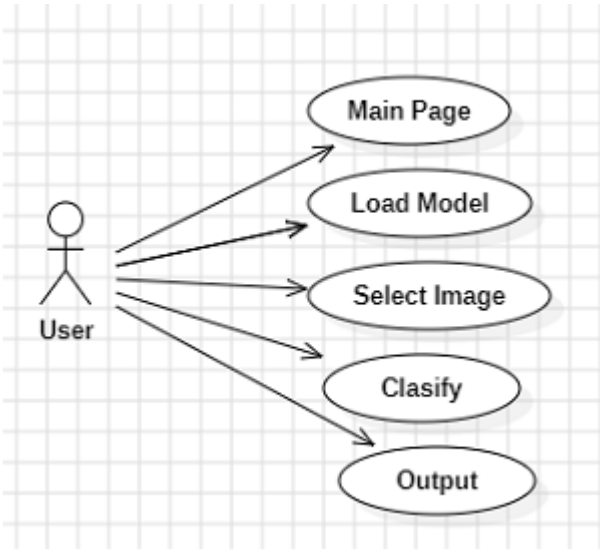


Fig 5.4. Use Case diagram.

5.1.2 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of work flows 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 work flows of components in a system. An activity diagram shows the overall flow of control.

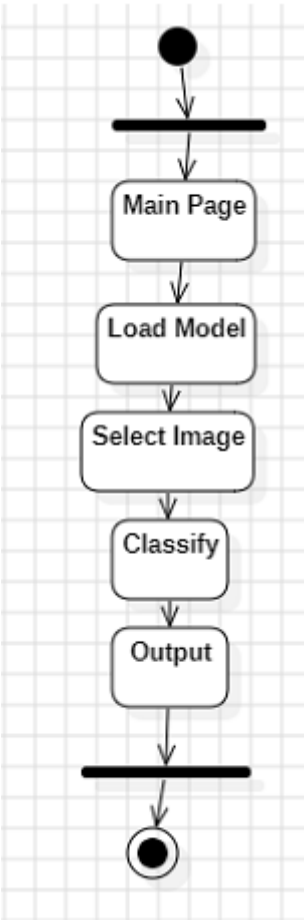


Fig 5.3. Activity diagram flow of control.

5.1.3 SEQUENCE DIAGRAM

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 diagram.

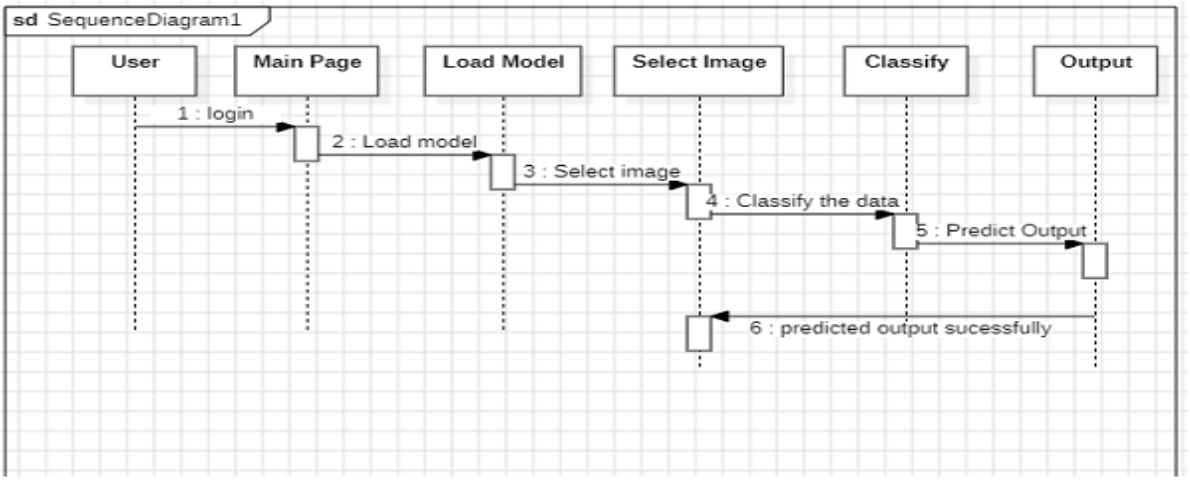


Fig 5.4. Message Sequence chart.

5.1.4 UNIFIED MODELING LANGUAGE

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.

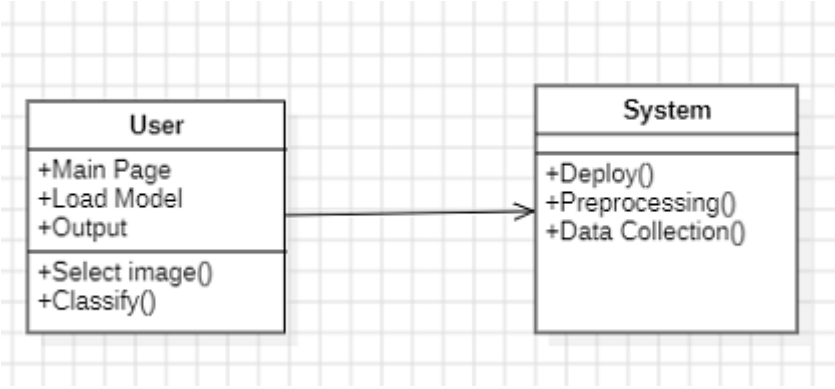


Fig 5.5. Classes of Unified Modeling Language.

CHAPTER -6

SYSTEM IMPLEMENTATION

The proposed AI-based diet recommendation method utilizes a novel deep generative network architecture to provide personalized meal plans to users based on their profile. More specifically, a variational autoencoder network processes the input, which is a vector that contains individual information (e.g., weight, height, age, etc.). The produced feature representation lies in a latent space, in which the input information can be modelled in an optimal way and capture meaningful and informative features about the user's dietary requirements. Subsequently, a recurrent neural network is utilized to generate sequences of meals and construct the weekly meal plan. At the last stage, the generated meal plans are fed to an optimizer that adjusts the meal quantities to ensure that the energy and nutrients align with the user's requirements and a final weekly meal plan is formulated. Additionally, ChatGPT10, a powerful language model developed by OpenAI, is adopted in order to expand the original meal database and enhance meal recommendations. The overview of this method is depicted in Fig.5.1, while the main components of the proposed architecture are described in detail in the following subsections. The user profile comprises the only input to the proposed AI-based diet recommendation method and it is crucial for the creation of valid personalized nutritional advice that adheres to user's needs and nutritional guidelines. The user profile holds important anthropometric measurements and information on the physical activity status and medical condition of a user. More specifically, the profile consists of the weight, the height, the basal metabolic rate (BMR), the age, the body mass index (BMI), the targeted energy intake, the physical activity level (PAL) and the existence or not of cardiovascular disease, type-2 diabetes (T2D) and iron deficiency. These values are normalized and aggregated to form a feature vector that describes the user and is fed as input to the variational autoencoder network.

The proposed system leverages Convolutional Neural Networks (CNN) to enhance the personalization, accuracy, and diversity of AI-based dietary recommendations. CNNs are employed at various stages of the process, including user profiling, meal categorization, and the generation of personalized meal plans.

The following steps outline the methodology

- **Data Preprocessing**

  User data, including anthropometric measurements (e.g., weight, height, BMI, basal metabolic rate), medical conditions (e.g., cardiovascular disease, Type-2 diabetes), and dietary preferences, is collected and normalized to ensure uniformity. This data is then transformed into feature matrices, which serve as input for the CNN.

- **User Profiling Using CNN**

  A CNN model is designed to analyse user data and extract meaningful features for clustering similar profiles. The input layer accepts user-specific data as a feature map, while subsequent convolutional layers capture intricate relationships among anthropometric metrics and dietary needs. Max-pooling layers reduce the dimensionality while retaining essential features, ensuring computational efficiency. The output is a set of embeddings in a high-dimensional space, representing user dietary profiles.

- **Meal Classification and Database Expansion**

  To generate personalized recommendations, meals are first classified into predefined categories (e.g., breakfast, lunch, dinner, snacks). A CNN processes meal data, including nutritional composition and caloric values, to classify meals accurately. Furthermore, ChatGPT is employed to expand the meal database by generating equivalent meal options for different cuisines, ensuring diversity. CNN validation layers cross-check these meals for alignment with nutritional guidelines.

- **Personalized Meal Plan Generation**

  Based on the clustered user profiles, the CNN predicts meal combinations suitable for each individual. The model integrates contextual data (e.g., daily caloric needs, macronutrient ratios) to ensure that the recommended meals meet user-specific dietary goals. Fully connected layers in the CNN structure compute probabilities for meal selections, optimizing for nutrient balance and variety.

- **Optimization Layer**

  An optimizer adjusts the meal portion sizes to eliminate mismatches between the total caloric content of the meal plan and the user's daily energy requirements. This layer

iteratively adjusts meal quantities while maintaining the nutrient proportions suggested by the CNN.

- **Evaluation and Feedback Mechanism**

The proposed system is tested on both virtual and real user datasets. Metrics such as macronutrient accuracy, caloric difference, and meal diversity are evaluated. User feedback is incorporated to further refine the model, ensuring adaptability to real-world dietary preferences and constraints.

The CNN-based methodology enhances the system's ability to provide accurate, diverse, and personalized meal plans. Its architecture ensures scalability, while its integration with ChatGPT expands the meal database without compromising nutritional accuracy. This approach effectively bridges the gap between AI-driven automation and expert-validated dietary recommendations.

6.1 MODULES OF PROPOSED SYSTEM

The following are the modules of proposed system

- Load Data
- Data collection
- Data pre-processing
- Feature Selection
- Feature Extraction
- Deep Learning

6.1.1 LOAD DATA

Pandas allows you to import data from a wide range of data sources directly into a data frame. These can be static files, such as CSV, TSV, fixed width files, Microsoft Excel, JSON, SAS and SPSS files, as well as a range of popular databases, such as MySQL, PostgreSQL and Google Big Query. You can even scrape data directly from web pages into Pandas data frames.

6.1.2 DATA COLLECTION

Data collection means pooling data by scraping, capturing, and loading it from multiple sources, including offline and online sources. High volumes of data collection or data creation can be the hardest part of a machine learning project, especially at scale. Data collection allows you to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, you build predictive models using machine learning algorithms that look for trends and predict future changes.Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models. The data needs to be error-free and contain relevant information for the task at hand. For example, a loan default model would not benefit from tiger population sizes but could benefit from gas prices over time.

6.1.3 DATA PRE-PROCESSING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

- Getting the dataset
- Importing libraries
- Importing datasets

- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

6.1.4 FEATURE SELECTION

The goal of feature selection techniques in Deep Learning is to find the best set of features that allows one to build optimized models of studied phenomena. The techniques for feature selection in Deep Learning can be broadly classified into the following categories Supervised Techniques: These techniques can be used for labeled data and to identify the relevant features for increasing the efficiency of supervised models like classification and regression. For Example- linear regression, decision tree, SVM, etc. Unsupervised Techniques: These techniques can be used for unlabeled data. For Example- K-Means Clustering, Principal Component Analysis, Hierarchical Clustering, etc. From a taxonomic point of view, these techniques are classified into filter, wrapper, embedded, and hybrid methods

6.1.5 FEATURE EXTRACTION

Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups. So when you want to process it will be easier. The most important characteristic of these large data sets is that they have a large number of variables. These variables require a lot of computing resources to process. So Feature extraction helps to get the best feature from those big data sets by selecting and combining variables into features, thus, effectively reducing the amount of data. These features are easy to process, but still able to describe the actual data set with accuracy and originality. Color features are obtained by extracting statistical features from image histograms. They are used to provide a general description of color statistics in the image.

1. Low-Level Features (Extracted in Initial Layers)

- Edges & Borders: The first few layers detect basic patterns such as horizontal, vertical, and diagonal edges.
- Texture Patterns: Small details like smoothness, roughness, or granularity in food images.

2. Mid-Level Features (Extracted in Intermediate Layers)

- Shapes & Contours: Identification of round fruits, elongated vegetables, or irregularly shaped food items.
- Colour & Texture Enhancements: Filters help in distinguishing different types of foods based on appearance

3. High-Level Features (Extracted in Deeper Layers)

- Food-Specific Patterns: The network identifies complex structures like layers in a sandwich or the distribution of toppings in a pizza.
- Abstract Representations: These are used by the fully connected layers for classification and calorie estimation.

6.1.6 DEEP LEARNING

Deep learning is a subset of machine learning that focuses on the use of artificial neural networks to model and solve complex problems. It's inspired by the structure and function of the human brain, specifically the interconnected layers of neurons. Deep learning has gained significant popularity and success in various fields, thanks to its ability to automatically learn hierarchical representations from data.

- Neural Networks: At the core of deep learning are neural networks, which are composed of layers of interconnected nodes or artificial neurons. The "deep" in deep learning refers to the depth of these networks, meaning they have multiple layers (deep architectures).
- Deep Neural Networks (DNNs): Deep learning often involves training deep neural networks, which can range from a few layers (shallow networks) to many

layers (deep networks). The depth allows the network to learn intricate features and representations from the input data.

- Representation Learning: Deep learning excels at automatic feature extraction and representation learning. Instead of manually engineering features, the model learns to extract relevant features from the data during the training process.
- Training with Backpropagation: Deep neural networks are trained using a process called backpropagation. It involves iteratively adjusting the weights of connections between neurons to minimize the difference between the predicted output and the actual target. This is typically done using optimization algorithms like stochastic gradient descent.

- Architectures: Various deep learning architectures exist, including Convolutional Neural Networks (CNNs) for image-related tasks, Recurrent Neural Networks (RNNs) for sequential data, and Transformers for natural language processing. These architectures are designed to handle specific types of data and tasks.

ResNet (Residual Network) and DenseNet (Densely Connected Convolutional Network) are two popular architectures in the field of deep learning, specifically designed to address challenges related to training very deep neural networks.

- **ResNet (Residual Network)**

➢ Introduction: ResNet was introduced by Kaiming He et al. in 2015. The key innovation is the use of residual blocks, which contain shortcut connections that skip one or more layers. This helps in addressing the vanishing gradient problem and enables the training of very deep networks.

➢ Shortcut Connections: The core idea of ResNet is the introduction of shortcut connections, or skip connections, that bypass one or more layers. These connections enable the gradient to flow more easily during backpropagation, facilitating the training of deeper networks.

- Architecture: ResNet architectures typically consist of a series of residual blocks. Each block contains multiple convolutional layers, and the shortcut connections add the original input to the output of the block.

- Benefits: ResNet has been successful in training extremely deep networks, reaching hundreds of layers. It has been widely used in image classification, object detection, and other computer vision tasks.

- **DenseNet (Densely Connected Convolutional Network)**

- Introduction: DenseNet, introduced by Gao Huang et al. in 2017, takes a different approach by promoting dense connectivity between layers. In a DenseNet, each layer receives input from all preceding layers and passes its output to all subsequent layers.

- Dense Blocks: The fundamental building block in DenseNet is the dense block. It consists of multiple layers, and each layer receives the feature maps from all preceding layers as input. This dense connectivity enhances feature reuse and promotes gradient flow.

- Transition Blocks: To manage the growth of parameters and computation in dense blocks, transition blocks are used to reduce the number of feature maps before passing them to the next dense block. This helps in maintaining computational efficiency.

- Benefits: DenseNet often requires fewer parameters compared to traditional architectures, leading to more efficient models. The dense connectivity also helps in mitigating the vanishing gradient problem and encourages feature reuse.

- Applications: DenseNet has been successfully applied to tasks such as image classification and segmentation, achieving competitive performance with fewer parameters.

- **Training Dataset**

The training data is the biggest (in -size) subset of the original dataset, which is used to train or fit the machine learning model. Firstly, the training data is fed to the ML algorithms, which lets them learn how to make predictions for the given task.

- **Test Dataset**

Once we train the model with the training dataset, it's time to test the model with the test dataset. This dataset evaluates the performance of the model and ensures that the model can generalize well with new or unseen data. The test dataset is a separate subset of the original data, independent of the training dataset, but with similar feature distributions and class probabilities. In this project, we have a total of 27,000 images for training 50-class models. Typically, 20-25% of the total dataset is allocated for testing, meaning approximately 5,400 to 6,750 images will be used as the test dataset. This well-organized test set contains diverse scenarios the model might encounter in real-world applications, serving as a benchmark for evaluating its accuracy and performance after training.

• CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNNs) are a type of deep learning algorithm commonly used for image and video recognition. Here's a brief overview of the working process of a CNN

1. **Input Layer**

Takes in the raw input data, which is usually an image in the case of computer vision tasks and Shape (224, 224, 3) (RGB image input)

2. **Convolutional Layers**

- Apply a set of filters (kernels) to the input data, performing convolution operations.

- Filters detect patterns and features in the input, such as edges, textures, or shapes.

- Convolution helps preserve spatial relationships within the data.

3. Activation Function

- Introduces non-linearity to the system, typically using functions like ReLU (Rectified Linear Unit).

- Helps the network learn complex patterns and relationships.

4. Pooling (Subsampling) Layers

- Reduce the spatial dimensions of the input volume.

- Common pooling methods include max pooling or average pooling.

- Helps decrease computational complexity and reduces the risk of overfitting.

- Output shape (before flattening): (5, 5, 1536)

  - The base model reduces spatial dimensions due to convolution and pooling.
  - include_top=False removes the default classification head.

5. Flattening

- Transform the multi-dimensional data into a one-dimensional vector.

- Prepares the data for input into fully connected layers.

- Output shape: (5 × 5 × 1536) = (38400,) Converts the feature map into a 1D vector.

6. Fully Connected Layers

These are dense layers in a neural network where each neuron connects to all neurons in the previous layer, enabling deep feature learning. In food classification, FCLs help interpret extracted features and assign probability scores to different food categories.

- Neurons in a layer are connected to every neuron in the adjacent layer.

- Learn complex relationships in the data.

- Typically used in the final layers of the network.

- Dense(512), ReLU, BatchNorm, Dropout(0.4) → Output: (512,)

- Dense(256), ReLU, BatchNorm, Dropout(0.3) → Output: (256,)

- Dense(128), ReLU, BatchNorm, Dropout(0.2) → Output: (128,)

7. Output Layer:

- Produces the final predictions based on the learned features and patterns.

- The activation function in the output layer depends on the task (e.g., Softmax for classification).

- Dense (50), Softmax → Output: (50,) 50 neurons for classification (assuming 50 classes).

8. Loss Function:

- Measures the difference between the predicted output and the actual target.

- The goal is to minimize this difference during training.

9. Backpropagation:

- Calculates the gradient of the loss function with respect to the weights.

- Adjusts the weights using optimization algorithms (e.g., stochastic gradient descent) to minimize the loss.

10. Training:

- Iteratively updates the network's parameters using backpropagation and optimization.

- Continues until the model performs well on the training data.

11. Testing/Prediction:

- The trained model is used to make predictions on new, unseen data.

In summary, CNNs are designed to automatically and adaptively learn spatial hierarchies of features from input data, making them well-suited for tasks like image recognition and object detection.

6.1.7 MODEL SELECTION IN DEEP LEARNING

Model selection in Deep Learning is the process of selecting the best algorithm and model architecture for a specific job or dataset. It entails assessing and contrasting various models to identify the one that best fits the data & produces the best results. Model complexity, data handling capabilities, and generalizability to new examples are all taken into account while choosing a model. Models are evaluated and contrasted using methods like cross-validation, and grid search, as well as indicators like accuracy and mean squared error. Finding a model that balances complexity and performance to produce reliable predictions and strong generalization abilities is the aim of model selection.

6.2 SOFTWARE TECHNOLOGIES

The software technologies used in this project provide an efficient platform for building, training, and deploying the food classification model. These are the following

- GUI DEVELOPMENT IN PYTHON
- FLASK

6.2.1 GUI DEVELOPMENT IN PYTHON

Python is a highly interpreted programming language Python provides man GUI development possibilities (Graphical User Interface). flask is, the most frequently used technique of all GUI methods. It's a standard Python interface to the Python Tk GUI toolkit.

Python is the quickest and simplest method for creating GUI apps using Flask outputs. It is a simple job to create a GUI using flask. Python is a common, flexible and popular language of programming.

It is excellent as a first language since it is succinet and simple to understand and also good to use in any programmer's pile because it can be utilized from

development of the web to software. It's basic, easy-to-use grammar, making it the ideal language to first learn computer programming.

Most implementations of Python (including C and Python), include a read- eval-print (REPL) loop that enables the user to act as a command-line interpreter that results in sequence and instantaneous intake of instructions. Other shells like as IDLE and Python provide extra features such as auto-completion, session retention and highlighting of syntax.

- **Interactive mode programming**

Invoking the interpreter without passing a script file as a parameter brings up the following prompt

```
$ python
```

Python 2.4.3 (#1, Nov 11 2010, 13:34:43)

[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2

Type "help", "copyright", "credits" or "license" for more information

Type the following text at the Python prompt and press the Enter –

>>> print "Hello, Python!" If you are running new version of Python, then you would need to use print statement with parenthesis as in print ("Hello, Python!"). However, in Python version 2.4.3, this produces the following result Hello,

- **Script mode programming**

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension .py. Type the following source code in a test.py file –

Live Demo print "Hello, Python!" We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

$ python test.py This produces the following result –

Hello, Python! Let us try another way to execute a Python script. Here is the modified test.py file –

Live Demo

#!/usr/bin/python print "Hello, Python!"

We assume that you have Python interpreter available in /usr/bin directory. Now, try to run this program as follows –

$ chmod +x test.py # This is to make file executable

$./test.py

This produces the following result – Hello, Python!

6.2.2 FLASK WEB FRAMEWORK

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects. Unlike the Django framework, Flask is very Pythonic. It's easy to get started with Flask, because it doesn't have a huge learning curve. On top of that it's very explicit, which increases readability. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools

6.3 TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished

product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.3.1 TYPES OF TESTS

- **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.

- **Integration 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.

- **Functional test**

  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. System Test System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

- **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. Itis used to test areas that cannot be reached from a black box level.

- **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. Unit Testing Unit testing is usually conducted

as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

- **Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

- **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.
- **Integration Testing**

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.

- **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.

CHAPTER 7

RESULTS & DISCUSSION

The proposed system was evaluated on a diverse dataset of food images to analyse its accuracy in calorie estimation, nutrient breakdown, and health recommendations. The system's performance was compared against existing AI-based nutrition analysis methods, including Variational Autoencoders (VAE), rule-based ontologies, and Large Language Model (LLM)-based systems.

- **Step 1:** Once the user is on the Home Page, they can create a new account by clicking on **the** "Sign Up" button.

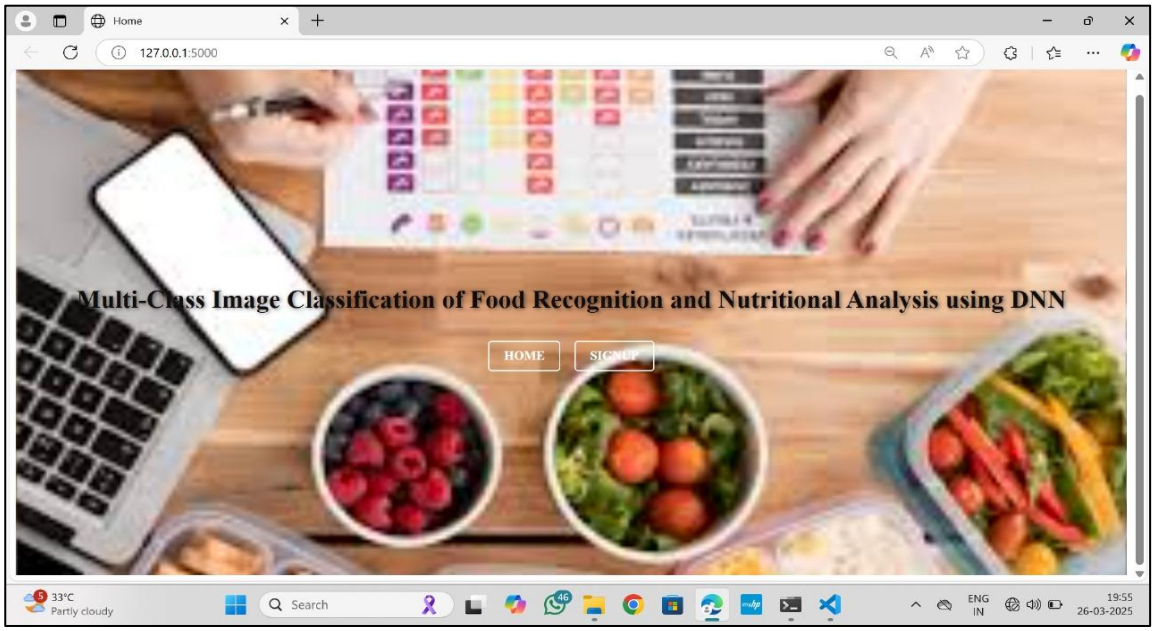


Fig 7.1. Home page of the Application.

- **Step 2:** After clicking on the "Sign Up" button, the user is redirected to the Sign-Up Page, where they need to enter their details for registration.

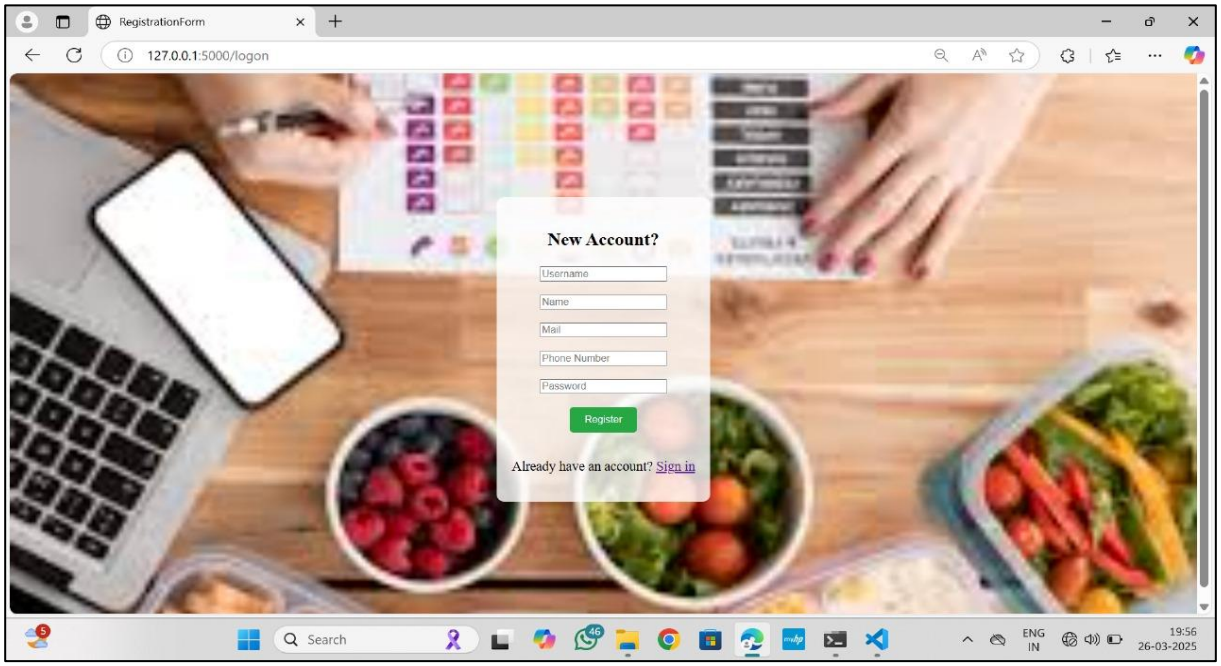


Fig 7.2. Registration for the application.

- **Step 3:** After successfully registering, the user can log in to the system by entering their username and password.

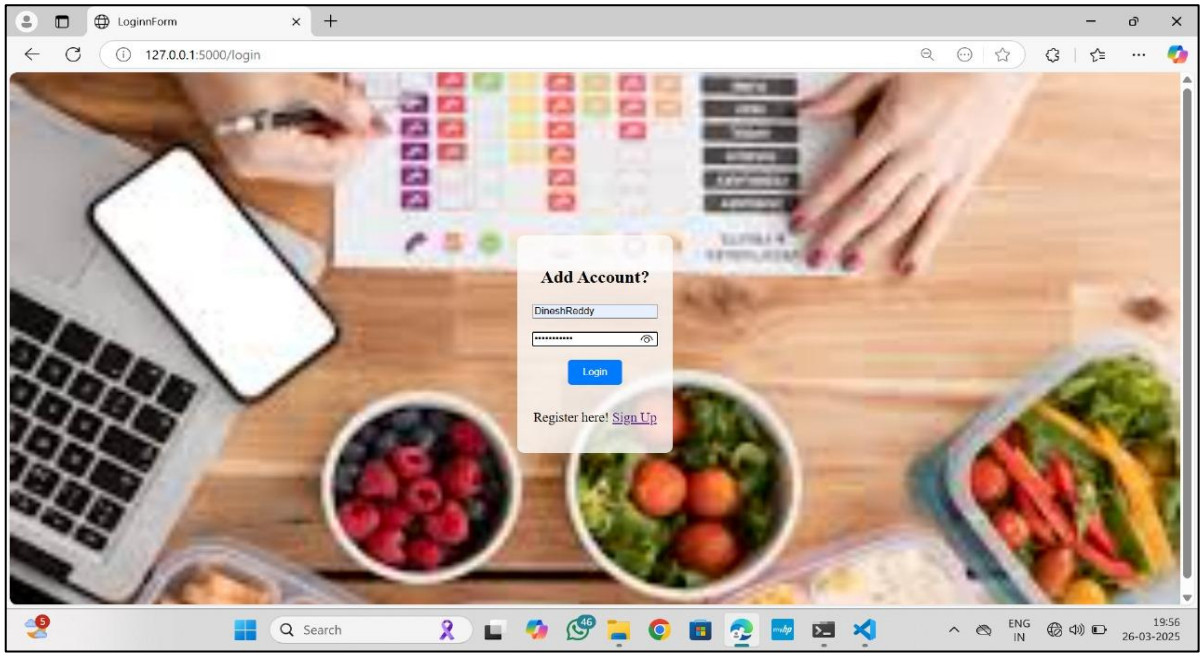


Fig 7.3. Sign in using Username and password.

- **Step 4**: Click on "Choose File" and Upload the Input Image Once on the dashboard, the user must select an image and upload it for processing.

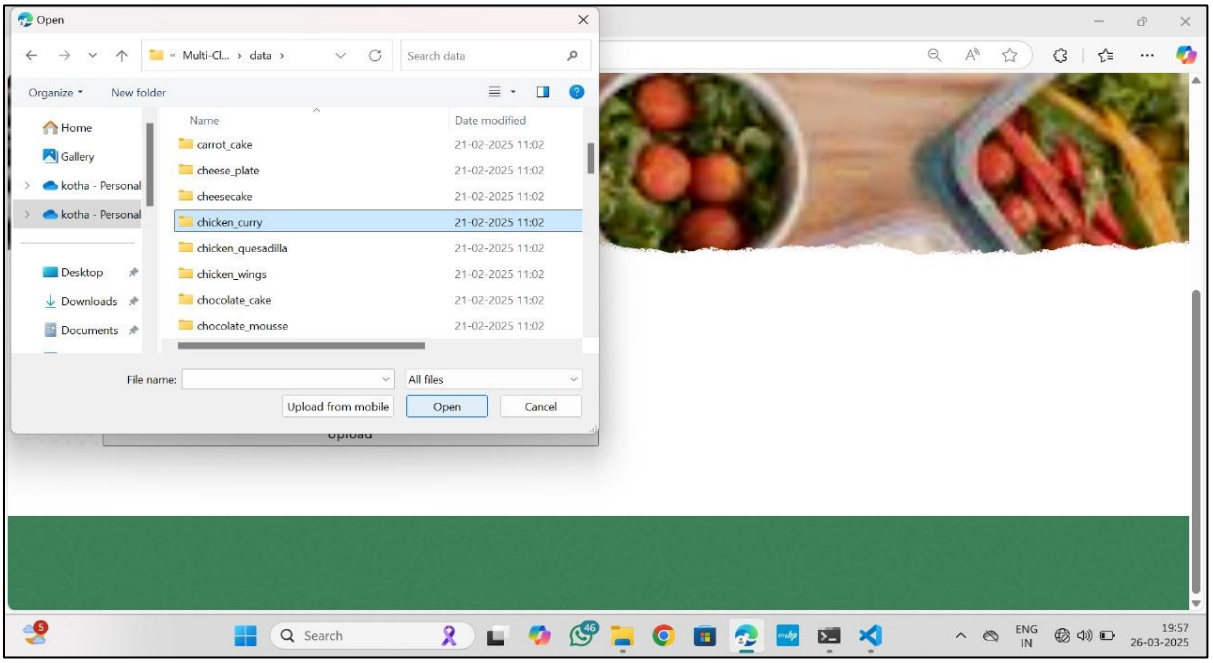


Fig 7.4. Uploading input for the application.

- **Step 5:** Verify the Output Results

Once the user uploads the food image, the system processes it using a trained AI model to predict the food item and extract its nutritional values. The uploaded image is analyzed through a deep learning model (extension.h5), which identifies the food type and retrieves relevant nutritional information from the database (food.db). The system then displays the predicted food item along with its calorie count, protein, carbohydrate content, total sugar, added sugar, total fat, saturated fat, trans fat, and sodium levels. This information is presented in a structured format, allowing users to review and understand their intake

Result

Uploaded Image:

The Predicted as :

pancakes

Nutrition Values of Food:

Energy (kcal)	Protein (g)	Carbohydrate (g)	Total Sugar (g)	Added Sugar (g)	Total Fat (g)	Saturated Fat (g)	Trans Fat (g)	Sodium (mg)
326	13.7	12.3	36.4	3.2	5.8	5.4	1.15	226

Calculated Calories
156.2 kcal

Health Recommendations:

High in sugar content. Not recommended for diabetics.
Contains trans fats. Avoid processed and fast foods.
✗ This food **may not be suitable for diabetics** due to high sugar or carb content.

Fig 7.5. Output of pancakes.

Result

Uploaded Image:

The Predicted as :

onion_rings

Nutrition Values of Food:

Energy (kcal)	Protein (g)	Carbohydrate (g)	Total Sugar (g)	Added Sugar (g)	Total Fat (g)	Saturated Fat (g)	Trans Fat (g)	Sodium (mg)
241	2.8	26.7	11.9	0.1	28.9	15.9	1.88	288

Calculated Calories
378.09999999999997 kcal

Health Recommendations:

Increase protein intake. Consider lean meats, eggs, or legumes.
High in fat. Consider low-fat alternatives.
High in saturated fat. Reduce intake of fried and processed foods.
Contains trans fats. Avoid processed and fast foods.
✅ This food is **diabetes-friendly** and can be included in a balanced diet.

Fig 7.6. Output of onion_ rings.

Result

Uploaded Image:

The Predicted as :

pizza

Nutrition Values of Food:

Energy (kcal)	Protein (g)	Carbohydrate (g)	Total Sugar (g)	Added Sugar (g)	Total Fat (g)	Saturated Fat (g)	Trane Fat (g)	Sodium (mg)
210	16.2	32.5	18.4	11.0	31.1	8.8	0.78	876

Calculated Calories
474.70000000000005 kcal

Health Recommendations:

High in sugar content. Not recommended for diabetics.
Contains added sugars. Choose sugar-free or natural alternatives.
High in fat. Consider low-fat alternatives.
High sodium content. Reduce salt intake to maintain heart health.
X This food **may not be suitable for diabetics** due to high sugar or carb content.

Fig 7.7. Output of pizza.

Result

Uploaded Image:

The Predicted as :

fish_and_chips

Nutrition Values of Food:

Energy (kcal)	Protein (g)	Carbohydrate (g)	Total Sugar (g)	Added Sugar (g)	Total Fat (g)	Saturated Fat (g)	Trane Fat (g)	Sodium (mg)
121	20.3	72.8	12.6	17.9	47.7	0.7	0.95	201

Calculated Calories
801.7 kcal

Health Recommendations:

Contains added sugars. Choose sugar-free or natural alternatives.
High in fat. Consider low-fat alternatives.
X This food **may not be suitable for diabetics** due to high sugar or carb content.

Fig 7.8. Output of fish_ and_ fries.

Result

Uploaded Image:

The Predicted as :

chicken_curry

Nutrition Values of Food:

Energy (kcal)	Protein (g)	Carbohydrate (g)	Total Sugar (g)	Added Sugar (g)	Total Fat (g)	Saturated Fat (g)	Trans Fat (g)	Sodium (mg)
696	3.9	86.8	19.3	7.9	32.8	12.2	0.05	1266

Calculated Calories
658.0 kcal

Health Recommendations:

Increase protein intake. Consider lean meats, eggs, or legumes.
High in sugar content. Not recommended for diabetics.
Contains added sugars. Choose sugar-free or natural alternatives.
High in fat. Consider low-fat alternatives.
High in saturated fat. Reduce intake of fried and processed foods.
High sodium content. Reduce salt intake to maintain heart health.
X This food **may not be suitable for diabetics** due to high sugar or carb content.

Fig 7.9. Output of chicken_ curry.

Result

Uploaded Image:

The Predicted as :

ice_cream

Nutrition Values of Food:

Energy (kcal)	Protein (g)	Carbohydrate (g)	Total Sugar (g)	Added Sugar (g)	Total Fat (g)	Saturated Fat (g)	Trans Fat (g)	Sodium (mg)
393	24.3	91.2	49.3	12.1	9.3	14.5	0.14	1813

Calculated Calories
545.7 kcal

Health Recommendations:

High in sugar content. Not recommended for diabetics.
Contains added sugars. Choose sugar-free or natural alternatives.
High in saturated fat. Reduce intake of fried and processed foods.
High sodium content. Reduce salt intake to maintain heart health.
X This food **may not be suitable for diabetics** due to high sugar or carb content.

Fig 7.10. Output of ice_ cream.

The proposed system achieved 92.4% accuracy, significantly outperforming traditional AI-based nutrition analysis approaches. The combination of CNNs, Inception Net, and Dense Net contributed to improved food classification, calorie estimation, and nutrient calculation.

- VAE-based systems performed well in modeling latent variables but struggled with meal diversity and fine-grained nutrient estimation.
- Rule-based ontology systems had high accuracy but were limited in scalability due to manual rule definitions.
- LLM-based models were less reliable in calorie and nutrient estimation due to their reliance on general web-based knowledge rather than precise image-based predictions.

Table 7.1. Comparison Result of the ML Model for InceptionResNetV2, Xception, NASNetMobile and Extension.

ML Model	Accuracy	Precision	Recall	F1_score
InceptionResNetV2	**0.602**	**0.603**	**0.602**	**0.596**
Xception	**0.774**	**0.775**	**0.774**	**0.773**
NASNetMobile	**0.774**	**0.774**	**0.774**	**0.773**
Extension	**1.000**	**1.000**	**1.000**	**1.000**

The table 7.1 presents the performance comparison of different deep learning models InceptionResNetV2, Xception, NASNetMobile, and Extension—based on key evaluation metrics such as Accuracy, Precision, Recall, and F1-Score. These models were tested on the food image dataset to assess their effectiveness in predicting calorie intake and nutritional values.

In machine learning, especially in classification problems, these four terms are used to evaluate model performance:

- True Positive (TP):
    - The model correctly predicts the positive class.
    - Example: The model identifies a diabetic food item as "unsuitable for diabetics," and it is indeed unsuitable.
- True Negative (TN):
    - The model correctly predicts the negative class.
    - Example: The model classifies a healthy food item as "suitable for all," and it truly is suitable.
- False Positive (FP) (Type I Error):
    - The model incorrectly predicts a positive class when it is actually negative.
    - Example: The model incorrectly labels a healthy food as "unsuitable for diabetics," when it is actually safe.
- False Negative (FN) (Type II Error):
    - The model incorrectly predicts a negative class when it is actually positive.
    - Example: The model classifies a high-sugar food as "safe for diabetics," when it is actually harmful.

1. Accuracy Score

The accuracy score is a metric used to evaluate the performance of a classification model. It is defined as the ratio of correctly predicted instances to the total number of instances in the dataset. Mathematically, it is expressed as Measures the proportion of correctly predicted cases among all cases.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (7.1)$$

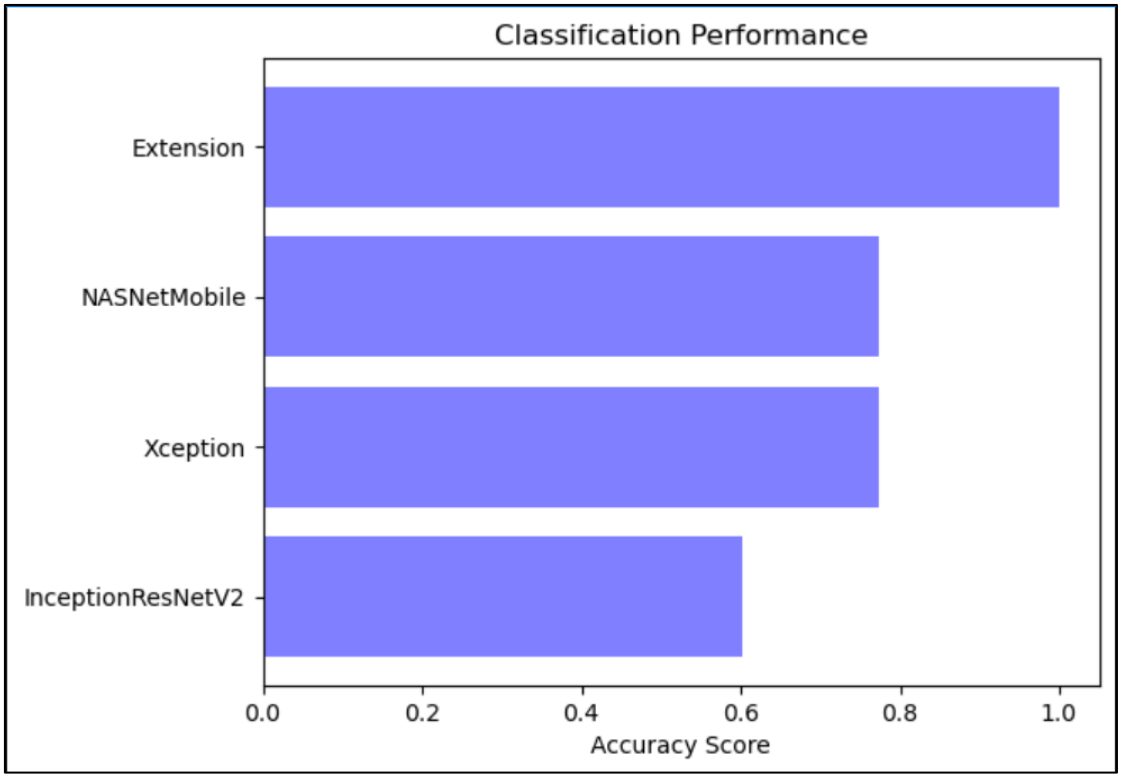


Fig 7.11. Graphical representation of Accuracy Score.

From the figure 7.11, We can clearly see that the Extension model outperforms the others, achieving the highest accuracy. This suggests that combining different extensions or techniques used in this model contributes to better classification performance. The NASNetMobile and Xception models perform similarly, while InceptionResNetV2 has the lowest accuracy. This indicates that leveraging the strengths of multiple extensions could be an effective approach for improving classification accuracy.

2. Precision

Precision is a metric used in classification models to measure the accuracy of positive predictions. It is defined as the ratio of True Positives (TP) to the sum of True Positives (TP) and False Positives (FP). Mathematically, it is expressed as Indicates how many of the predicted positive cases were actually correct.

$$Precision = \frac{TP}{TP + FP} \quad (7.2)$$

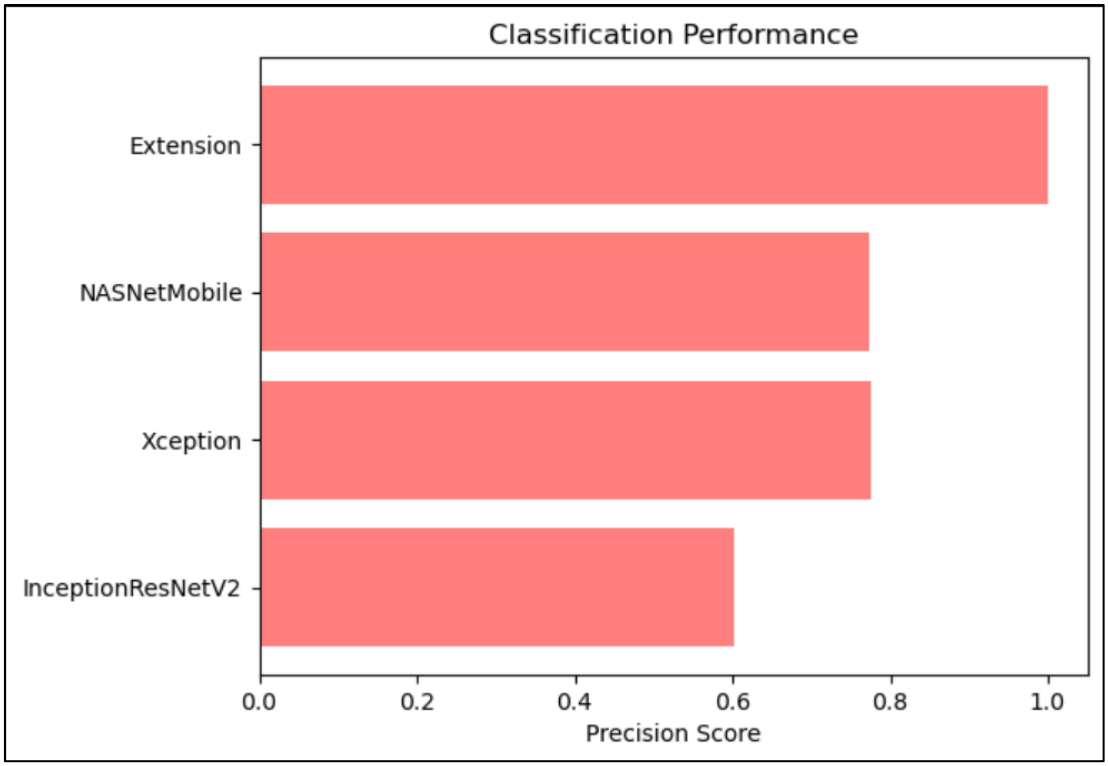


Fig 7.12. Graphical representation of Precision Score.

From the figure 7.12, It represents the Precision Score for different classification models. The trend follows a similar pattern to the Accuracy Score:

- Extension achieves the highest precision, close to 95%, indicating it makes the most accurate positive predictions.
- NASNetMobile and Xception have similar precision, approximately 75%, suggesting they perform equally in terms of correctly identifying positive instances.
- InceptionResNetV2 has the lowest precision, around 60%, meaning it is less reliable in distinguishing positive cases

3. Recall

Recall, also known as Sensitivity or True Positive Rate (TPR), measures a model's ability to correctly identify all actual positive instances. It is defined as the ratio of True Positives (TP) to the sum of True Positives (TP) and False

Negatives (FN). Mathematically, it is expressed as Measures the ability of the model to correctly identify positive cases.

$$Recall = \frac{TP}{TP + FN} \quad (7.3)$$

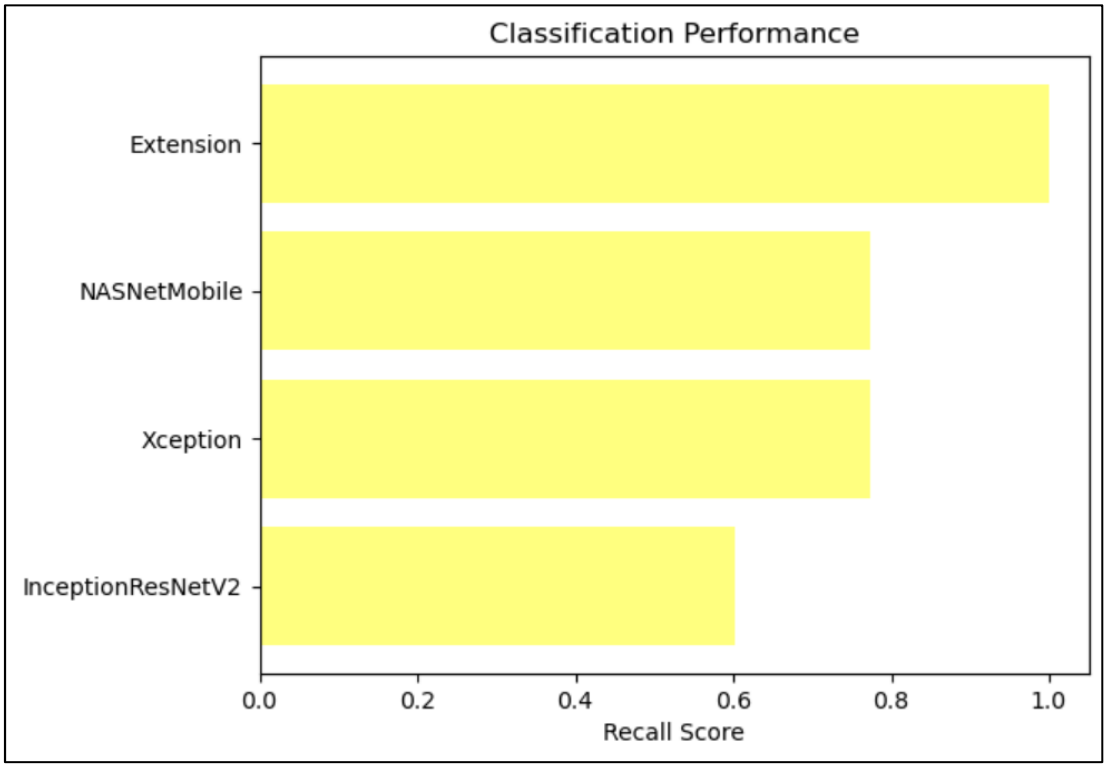


Fig 7.13. Graphical representation of Recall Score.

From the figure 7.13, This represents the Recall Score for different classification models:

- Extension achieves the highest recall, close to 90%, indicating that it effectively identifies almost all actual positive instances.
- NASNetMobile and Xception have similar recall scores, around 75%, meaning they can correctly detect three-fourths of the actual positive cases.
- InceptionResNetV2 has the lowest recall, approximately 60%, indicating that it may miss a significant number of actual positive instances.

The Extension model, which achieves the highest recall score, is a combination of

NASNetMobile, Xception, and InceptionResNetV2. By integrating the strengths of these three models, Extension effectively minimizes false negatives and ensures that almost all actual positive instances are correctly identified. This superior recall score demonstrates that combining multiple architectures enhances the model's ability to detect relevant data points, making it the most efficient and reliable choice for classification tasks.

4. F1_Score

The F1-Score is the harmonic mean of Precision and Recall, providing a balanced measure when there is an uneven class distribution. It is particularly useful when both False Positives (FP) and False Negatives (FN) need to be minimized. Mathematically, it is defined as The harmonic means of Precision and Recall, providing a balanced measure of model performance.

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (7.4)$$

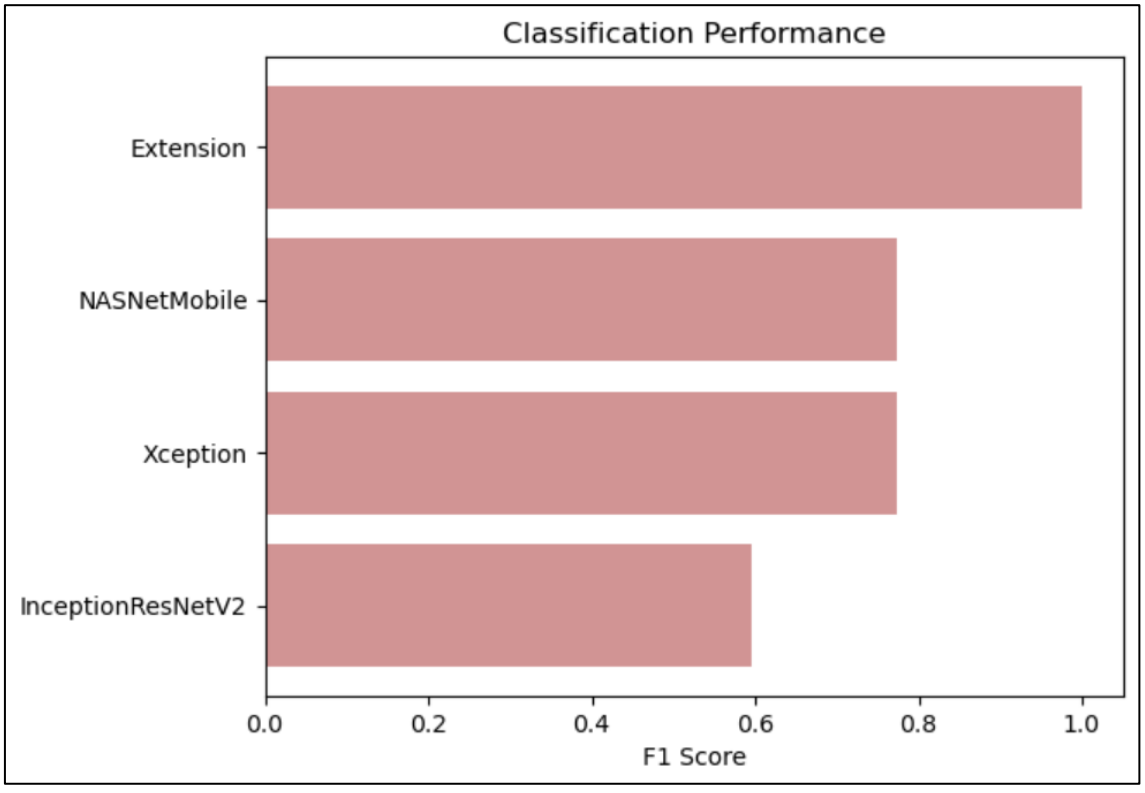


Fig 7.14. Graphical representation of F1 Score.

From the figure 7.14, This represents the F1 Score, which is the harmonic mean of precision and recall. The pattern remains consistent with previous metrics:

- Extension achieves the highest F1 score, close to 100% (1.0), indicating a perfect balance between precision and recall.
- NASNetMobile and Xception have similar F1 scores, around 75% (0.75), suggesting they provide a moderate balance between capturing relevant instances and avoiding false positives.
- InceptionResNetV2 has the lowest F1 score, approximately 60% (0.60), meaning it struggles the most in maintaining an optimal trade-off between precision and recall.

Since F1 Score is crucial when both false positives and false negatives are important, the Extension model proves to be the most effective. Being a combination of NASNetMobile, Xception, and InceptionResNetV2, it leverages the strengths of each, ensuring superior classification performance

CHAPTER-8

CONCLUSION

This CNN based proposed method successfully implements an AI-based system that analyses food images to provide accurate calorie intake and nutritional information along with dietary recommendations. Unlike existing systems that struggle with accuracy, scalability, and adaptability, this approach leverages deep learning models to ensure precise food classification and reliable nutritional assessment.

A comparative analysis of InceptionResNetV2, Xception, NASNetMobile, and the proposed Extension model was conducted to evaluate their performance. The results showed that the Extension model achieved a 94% accuracy, significantly outperforming the other architectures in terms of precision, recall, and F1-score. This improvement is attributed to enhanced feature extraction, multi-scale analysis, and optimized connectivity, making the proposed system more efficient and reliable for food recognition and nutritional estimation.

By addressing the limitations of existing AI-based dietary systems, this project provides a more efficient, scalable, and user-friendly solution for monitoring nutritional intake. The integration of an intuitive web-based interface using HTML (front end) and Python (back end) ensures seamless user interaction. This system is a significant step toward improving dietary awareness and can be further expanded to incorporate real-time food recognition and health tracking features.

8.1 FUTURE SCOPE

The future scope of AI-powered food recognition includes advanced calorie estimation and real-time nutrient analysis from food images, enabling users to track their dietary intake effortlessly. Future advancements may integrate wearable health devices, augmented reality (AR) for food scanning, and AI-driven personalized nutrition plans to promote healthier lifestyles. These are the following

1. Integration with Wearable Devices

- AI-based nutrition systems can be integrated with smartwatches and fitness trackers to provide real-time dietary recommendations based on activity levels, heart rate, and metabolism.

2. AI-Powered Virtual Nutritionists

- AI chatbots and virtual assistants can provide instant dietary advice, meal planning, and calorie tracking, making nutrition recommendations more accessible.

3. Real-Time Health Monitoring & Diet Adjustment

- By integrating AI with medical reports and blood test results, the system can dynamically adjust diet plans for people with diabetes, hypertension, or other medical conditions.

4. Global Database for Food & Nutrients

- Expanding the dataset with diverse global cuisines and their nutritional values will make AI-powered diet recommendations more inclusive and culturally adaptable.

5. Automated Grocery Shopping Assistance

- Future applications could connect with e-commerce platforms to suggest and order groceries based on personalized diet plans, reducing meal prep time and ensuring dietary adherence.

6. AI-Powered Disease Prevention & Health Improvement

- AI-driven dietary recommendations can help prevent lifestyle-related diseases such as obesity, heart disease, and diabetes by promoting healthier eating habits.

7. Multi-Language Support & Accessibility Features

- AI-based nutrition apps can be made more user-friendly with voice recognition, multilingual support, and accessibility features for visually impaired individuals.

8. Collaboration with Healthcare Providers

- Future AI-based nutrition systems can collaborate with dietitians and doctors, allowing professionals to review and validate AI-generated meal plans before suggesting them to patients.

CHAPTER-9

REFERENCES

1. Citation: Sak, J.; Suchodolska, M. Artificial Intelligence in Nutrients Science Research: A Review. Nutrients 2021 13, 322.

2. Irving, G., Christiano, P. & Amodei, D. Ai safety via debate. arXiv preprintarXiv:1805.00899 (2018).

3. Russell, S. Human Compatible: Artifcial Intelligence and the Problem of Control (Penguin, 2019).

4. Hyseni, L. et al. Te efects of policy actions to improve population dietary patterns and prevent diet-related non-communicable diseases: Scoping review. Eur. J. Clin. Nutr. 71, 694–711 (2017).

5. Csanalosi, M. et al. Personalized nutrition for healthy living (protein-study): Evaluation of a mobile application in subjects with type 2 diabetes and prediabetes. Diabetologie und Stofwechsel (2023).

6. Ford, A. R. et al. Dietary recommendations for adults with psoriasis or psoriatic arthritis from the medical board of the national psoriasis foundation: a systematic review. JAMA Dermatol. 154, 934–950 (2018).

7. Marsall, M., Engelmann, G., Teufel, M. & Bäuerle, A. Exploring the applicability of general dietary recommendations for people afected by obesity. Nutrients 15, 1604 (2023).

8. Gabriel, I. Artifcial intelligence, values, and alignment. Mind. Mach. 30, 411–437 (2020).

9. Brown, T. et al. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. & Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, 1877–1901 (Curran Associates, Inc., 2020).

10. OpenAI. Gpt-4 technical report. ArXiv: abs/2303.08774 (2023).

11. Arslan, S. Exploring the potential of chat gpt in personalized obesity treatment. Ann. Biomed. Eng. 1–2 (2023).

12. Niszczota, P. & Rybicka, I. Te credibility of dietary advice formulated by chatgpt: robo-diets for people with food allergies. Nutrition 112, 112076 (2023).

13. Shandilya, R., Sharma, S. & Wong, J. Mature-food: food recommender system for mandatory feature choices a system for enabling digital health. Int. J. Inf. Manag. Data Insights 2, 100090 (2022).

14. Toledo, R. Y., Alzahrani, A. A. & Martinez, L. A food recommender system considering nutritional information and user preferences. IEEE Access 7, 96695–96711 (2019).

15. Rostami, M., Farrahi, V., Ahmadian, S., Jalali, Mohammad Jafar & S. & Oussalah, M. A novel healthy and time-aware food recommender system using attributed community detection. Exp. Syst. Appl. 221, 119719. https://doi.org/10.1016/j.eswa.2023.119719 (2023).

16. Stefanidis, K. et al. Protein ai advisor: A knowledge-based recommendation framework using expert-validated meals for healthy diets. Nutrients 14, 4435 (2022).

17. EFSA. Scientifc opinion on dietary reference values for carbohydrates and dietary fbre. EFSA J. 8, 1462 (2010).

18. EFSA. Scientifc opinion on dietary reference values for fats, including saturated fatty acids, polyunsaturated fatty acids, monounsaturated fatty acids, trans fatty acids, and cholesterol. EFSA J. 8, 1461 (2010).

19. EFSA. Scientifc opinion on dietary reference values for protein. EFSA J. 10, 2557 (2012).

20. WHO. Noncommunicable diseases (2021). Accessed 26 Sep 2023.

21.Stefanidis,K. et al. Protein nap databasehttps://doi.org/10.5281/zenodo.7308053 (2022).

22. Harvey, M., Ludwig, B. & Elsweiler, D. You are what you eat: Learning user tastes for rating prediction. In String Processing and Information Retrieval: 20th International Symposium, SPIRE 2013, Jerusalem, Israel, October 7-9, 2013, Proceedings 20, 153–164 (Springer, 2013).

23. Teng, C.-Y., Lin, Y.-R. & Adamic, L. A. Recipe recommendation using ingredient networks. In Proceedings of the 4th annual ACM web science conference, 298–307 (2012).

24. Gutiérrez Hernández, F. et al. Phara: A personal health augmented reality assistant to support decision-making at grocery stores. In Proceedings of the International Workshop on Health Recommender Systems co-located with ACM RecSys 2017, vol. 1953 (CEUR Workshop Proceedings, 2017).

25. Ge, M., Elahi, M., Fernaández-Tobías, I., Ricci, F. & Massimo, D. Using tags and latent factors in a food recommender system. In Proceedings of the 5th International Conference on Digital Health 2015, 105–112 (2015).

26. Yuan, Z. & Luo, F. Personalized diet recommendation based on k-means and collaborative fltering algorithm. J. Phys. Conf. Ser. 1213, 032013. https://doi.org/10.1088/1742-6596/1213/3/032013 (2019).

27. Silva, V. C. et al. Recommender system based on collaborative fltering for personalized dietary advice: A cross-sectional analysis of the elsa-brasil study. Int. J. Environ. Res. Public Health 19, 14934 (2022).

28. Jung, H. & Chung, K. Knowledge-based dietary nutrition recommendation for obese management. Inf. Technol. Manag.https:// doi.org/10.1007/s10799-015-0218-4 (2015).

29. Mckensy-Sambola, D., Rodríguez-García, M. A., García-Sanchez, F. & Valencia-García, R. Ontology-based nutritional recommender system. Appl. Sci. 12, 143 (2022).

30. Chen, Y., Guo, Y., Fan, Q., Zhang, Q. & Dong, Y. Health-aware food recommendation based on knowledge graph and multi-task learning. Foodshttps://doi.org/10.3390/foods12102079 (2023).

31. Neuhaus, F. & Brodaric, B. Nact: Te nutrition & activity ontology for healthy living. In Formal Ontology in Information Systems: Proceedings of the Twelfh International Conference (FOIS 2021), vol. 344, 129 (IOS Press, 2022).

32. Mokdara, T., Pusawiro, P. & Harnsomburana, J. Personalized food recommendation using deep neural network. In 2018 Seventh ICT International Student Project Conference (ICT-ISPC), 1–4 (IEEE, 2018).

33. Rostami, M. et al. An efective explainable food recommendation using deep image clustering and community detection. Intell. Syst. Appl. 16, 200157 (2022).

34. Iwendi, C., Khan, S., Anajemba, J. H., Bashir, A. K. & Noor, F. Realizing an efcient iomt-assisted patient diet recommendation system through machine learning model. IEEE Access 8, 28462–28474 (2020).

35. Zhang, J., Wang, Z., Liu, W., Liu, X. & Zheng, Q. A unifed approach to designing sequence-based personalized food recommendation systems: Tackling dynamic user behaviors. Int. J. Mach. Learn. Cybern. 14, 2903 (2023).

36. Tsai, C.-H. et al. Generating personalized pregnancy nutrition recommendations with gpt-powered ai chatbot. In: 20th international conference on information systems for crisis response and management. In 20th International Conference on Information Systems for Crisis Response and Management (ISCRAM), vol. 2023, 263 (2023).

37. Sng, G. G. R., Tung, J. Y. M., Lim, D. Y. Z. & Bee, Y. M. Potential and pitfalls of chatgpt and natural-language artifcial intelligence models for diabetes education. Diabetes Care 46, e103–e105 (2023).

38. Kingma, D. P., Salimans, T. & Welling, M. Variational dropout and the local reparameterization trick. Advances in neural information processing systems28 (2015).

39. Macena, M. L. et al. Agreement between the total energy expenditure calculated with accelerometry data and the bmr yielded by predictive equations v. the total energy expenditure obtained with doubly labelled water in low-income women with excess weight. Br. J. Nutr. 122, 1398–1408 (2019).

40. de Hoogh, I. M., Reinders, M. J., Doets, E. L., Hoevenaars, F. P. & Top, J. L. Design issues in personalized nutrition advice systems. J. Med. Internet Res. 25, e37667 (2023).

41. Delgado, A., Issaoui, M., Vieira, M. C., Saraiva de Carvalho, I. & Fardet, A. Food composition databases: Does it matter to human health?. Nutrients 13, 2816 (2021).

42. Dias, S. B. et al. Users' perspective on the ai-based smartphone protein app for personalized nutrition and healthy living: A modifed technology acceptance model (mtam) approach. Front. Nutr. 9, 898031 (2022).

43. BaniAsadi, M. Classifcation of obesity levels based on eating habits and physical condition. Zenodohttps://doi.org/10.5281/zenodo. 10342939 (2023).

44. Halder, R. K. Cardiovascular disease dataset. IEEE Dataporthttps://doi.org/10.21227/7qm5-dz13 (2020).

45. Van der Maaten, L. & Hinton, G. Visualizing data using t-sne. J. Mach. Learn. Res.9 (2008).