Diet Planner

**A PROJECT REPORT**

***Submitted by***

ANSHIKA THAKUR – (21CBS1065)

***in partial fulfillment for the award of the degree of***

**COMPUTER SCIENCE ENGINEERING WITH BUSINESS SYSTEM**

**IN**

AIT-CSE



**Chandigarh University**

April 2024



**BONAFIDE CERTIFICATE**

Certified that this project report **“Diet Planner”** is the bonafide work of “Anshika (21CBS1065)**”** who carried out the project work under my/our supervision.

**SIGNATURE**

**(HOD)**

**(CSBS DEPARTMENT)**

**SIGNATURE**

**(SUPERVISOR)**

**(CSBS DEPARTMENT)**

Submitted for the project viva-voce examination held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**TABLE OF CONTENTS**

[Abstract](#_bookmark2) 7

[Graphical Abstract ...](#_bookmark3) 8

[Chapter 1 Introduction](#_TOC_250014) 10

1.1 [Problem Definition](#_TOC_250011) 10

1.2 [Project Overview](#_TOC_250010) 11

1.3 [Hardware Specification](#_TOC_250009) 12

1.4 [Software Specification](#_TOC_250008) 13

[Chapter 2 Background and Alternatives](#_TOC_250013) 15

2.1 [Prolog](#_TOC_250011) 15

2.2 [Diet Planner](#_TOC_250010) 15

2.3 [Types](#_TOC_250009) 16

2.4 Alternatives17

[Chapter 3 Problem Formulation](#_TOC_250012) 18

3.1 [Key elements of the issue](#_TOC_250011) 18

4.2 [Proposed Solution](#_TOC_250010) 18

[Chapter 4 Objectives](#_TOC_250007) 20

[Chapter 5 Methodologies](#_TOC_250002) 21

5.1 [Phases](#_TOC_250011) 21

5.2 [Timeline](#_TOC_250010) 23

5.3 [Use Case](#_TOC_250010) 23

5.4 [Code](#_TOC_250010) 24

[Chapter 6 Experimental Setup](#_TOC_250001) 40

[Chapter 7 Results and Discussion](#_TOC_250001) 42

7.1 [System Performance Evaluation](#_TOC_250011) 42

7.2 [Output](#_TOC_250010) 44

[Chapter 9 Conclusion](#_TOC_250001) 49

[Chapter 10 Future Scope](#_TOC_250002) 50

[Tentative chapter plan for proposed work](#_TOC_250000) 52

References 53

## List of Figures

**Figure i …………………………………………………………………………………….Graphical Abstract**

**Figure 5.1 ……………………………………………………………………………….Timeline Gantt Chart**

**Figure 5.2 ……………………………………………………………………………………………..Use Case**

**Figure 7.1 ………………………………………………………………………………Performance Metrics**

**Figure 7.2 ……………………………………………………………………………………………...Output**

**Figure 7.3 …………………………………………………………………………….……………….Output**

**Figure 7.4 ……………………………………………………………………………………****………Output**

## List of Tables

**Table 5.1 ………………………………………………………………………………….Timeline**

**Table 7.1 ………………………………………………………………………………….Metrics Table**

# ABSTRACT

This project is a complete diet planner system designed to encourage people to lead better lifestyles and make meal planning easier for them. The system, which makes use of a large database of food products classified by kind, nutritional value, and appropriate mealtime, is implemented using Prologue.

The diet planner provides customised meal plans based on nutritional needs and personal preferences. The system creates daily meal plans for users based on variables including calorie consumption, nutritional balance, and vegetarian or non-vegetarian choices. It also gives customers comprehensive nutritional data for every meal, enabling them to make educated diet choices.

The project guarantees accuracy and dependability in meal suggestions by means of thorough testing and optimisation. The straightforward and user-friendly design of the user interface improves accessibility for a variety of users. Additionally, by taking into account a range of dietary choices and limits, the method promotes inclusion.

The diet planner enhances general wellbeing by encouraging better eating practices and streamlining meal preparation. The project's focus on accessibility and user-centric design is in line with the overarching objective of enabling people to make better lifestyle decisions.

**Keywords:**

Personalised meal plans, Vegetarian and Non-Vegetarian diet planners, Prologue, meal planning, nutritional content, accessibility, inclusivity, and a healthier lifestyle.

# GRAPHICAL ABSTRACT

This graphical abstract illustrates the flow of our Diet Planner. The components include:

1. **Start**

- Open prolog file

- Enter file name

- Enter the query : accept\_user-input

**2. User Input:**

- Prompt user for their name, age, vegetarian preference, weight, and height.

- Read user's responses.

**3. Calculate BMI:**

- Calculate the Body Mass Index (BMI) using the provided weight and height.

- BMI = Weight / (Height^2)

**4. Categorize BMI:**

Categorize the BMI into "underweight," "normal weight," or "overweight" based on predefined thresholds.

1. **Generate Weekly Diet Schedule:**

- If the user is vegetarian:

- Generate a weekly diet schedule for a vegetarian based on the predefined meal options.

- If the user is not vegetarian:

- Generate a weekly diet schedule for a non-vegetarian based on the predefined meal options.

1. **Print Weekly Diet Schedule:**

- Print the weekly diet schedule showing meals for each day of the week.

- Display total calories, protein, carbs, and fats for each day.

1. **End**

This graphical abstract outlines the process of the meal planning system, starting from user input to generating and printing the weekly diet schedule based on the user's preferences and BMI category.

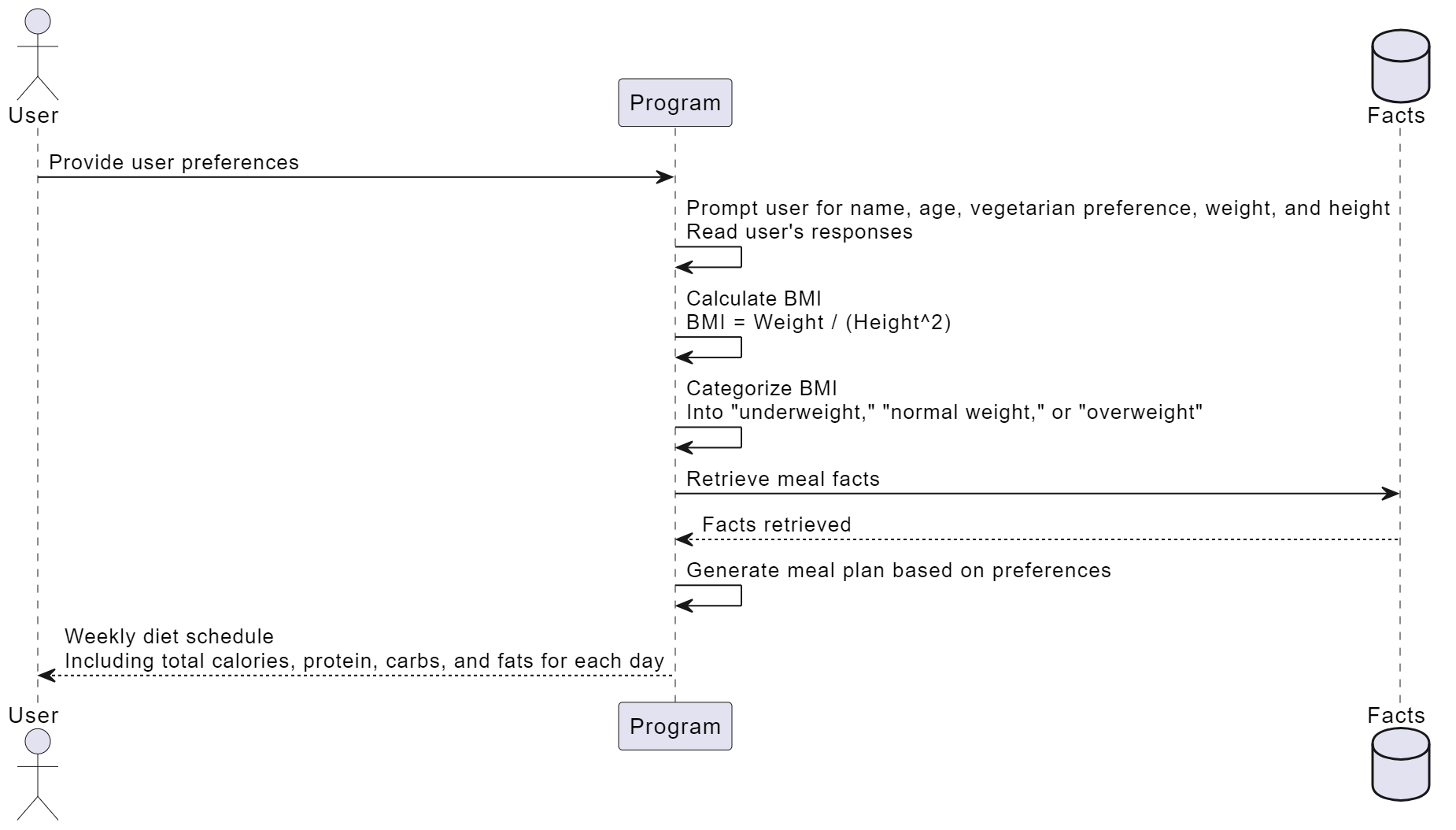


Figure i

# Chapter 1: Introduction

# In the era of rapidly advancing technology, the imperative to foster accessibility and inclusivity stands as a cornerstone of progress. This project embarks on a mission to contribute significantly to this endeavor by introducing an innovative dietary planning system aimed at empowering individuals to make healthier lifestyle choices. Embracing state-of-the-art programming techniques such as Prolog, the system not only symbolizes technological advancement but also epitomizes a dedication to user-centric design and inclusivity.

# The basis of this research is the creation of an approved nutrition plan. The system attempts to use a large database of food items broken down by nutritional values, dietary restrictions, and social times to provide meal plans based on personal preferences and health goals. The combination of different factors such as health goals, dietary intake and dietary restrictions ensures that each meal plan is carefully tailored to the client's needs.

# In addition to core functionality, Meal Planner includes a variety of features designed to improve user experience and accessibility. From user insights to quality nutritional information, the system is designed to provide users with the information and tools they need to make healthy nutrition decisions. The system encourages participation by tracking different dietary preferences and restrictions, ensuring its work benefits all users.

# As technology continues to permeate every aspect of daily life, the importance of accessibility and user-friendly design cannot be ignored. Combining technological tools with a commitment to integration, the program aims to not only provide practical solutions for meal preparation, but has also set a precedent for the development of health and wellness tools. Through rigorous testing and adherence to the highest practical standards, the project aims to deliver powerful and effective nutrition plans that enable people to take control of their health and wellness. It is a technological breakthrough and the next step in the ongoing quest to develop an accessible and inclusive digital environment. The following sections of this report will delve more deeply into the process, findings, and outcomes of this effort, exploring the entire journey of creating sustainable wellness.

## 1.1 Problem Definition

Despite the growing awareness of the importance of nutrition and healthy eating habits, many individuals struggle to plan and maintain balanced diets. Factors such as busy schedules, lack of nutritional knowledge, and dietary restrictions often hinder efforts to make informed food choices. As a result, people may resort to convenient but less nutritious options, leading to suboptimal health outcomes.

Moreover, the abundance of conflicting information and fad diets in the media further complicates the task of navigating dietary choices. Many individuals find it challenging to sift through this information overload and discern scientifically sound nutritional advice.

In this context, there is a clear need for a reliable and user-friendly solution that simplifies meal planning, provides accurate nutritional information, and promotes healthier eating habits. Such a solution should cater to diverse dietary preferences and restrictions while offering personalized recommendations tailored to individual needs.

By addressing these challenges, the proposed diet planner project seeks to empower users to make informed dietary decisions, improve their overall nutritional intake, and ultimately enhance their well-being.

## 1.2 Problem Overview

In today's fast-paced lifestyle, maintaining a healthy diet and making informed food choices can be challenging for many individuals. Factors such as busy schedules, limited culinary knowledge, and conflicting dietary information often contribute to poor eating habits and nutritional imbalances. Additionally, dietary preferences and restrictions, such as vegetarianism or food allergies, further complicate the process of meal planning.

Traditional methods of meal planning, relying on manual calculations and generic nutritional guidelines, may not adequately address the diverse needs and preferences of individuals. Moreover, the lack of personalized guidance and detailed nutritional information can hinder individuals' ability to make optimal food choices.

This project aims to tackle these challenges by developing a sophisticated diet planner system that provides personalized meal recommendations and detailed nutritional analysis. By leveraging the power of artificial intelligence and a comprehensive database of food items, the system aims to empower users to make informed decisions about their diet while accommodating their dietary preferences and restrictions.

Through the implementation of advanced algorithms and user-friendly interfaces, the project seeks to streamline the process of meal planning and promote healthier eating habits among individuals. By addressing the root causes of poor dietary choices and providing tailored solutions, the diet planner aims to contribute to improved overall health and well-being for users.

Key challenges to address in developing the diet planner include:

**i. Building a comprehensive database:** Curating a vast database of food items with accurate nutritional information, including calorie counts, macronutrient composition, and mealtime suitability.

**ii. Personalization and customization:** Developing algorithms to generate personalized meal plans based on individual preferences, dietary requirements, and health goals.

**iii. Accuracy and reliability:** Ensuring that the system's recommendations are accurate, reliable, and aligned with nutritional guidelines and dietary best practices.

**iv. Accessibility and inclusivity:** Incorporating features and functionalities that cater to diverse dietary preferences, cultural backgrounds, and health conditions, ensuring accessibility and inclusivity for all users.

By addressing these challenges, the project aims to deliver a robust and effective solution that empowers individuals to take control of their diet and make healthier choices for a better quality of life.

## 1.3 Hardware Specification

For a project like the diet planner system implemented in Prolog, the hardware specifications required are relatively modest since Prolog is not resource-intensive compared to some other programming languages. Here's a basic hardware specification:

1. **Processor:** A modern multi-core processor, such as an Intel Core i5 or AMD Ryzen 5, would suffice. Prolog doesn't require a high-end processor, but having multiple cores can help in multitasking if you're running other applications alongside the diet planner.
2. **Memory (RAM):** A minimum of 4GB RAM is recommended to ensure smooth operation of the diet planner. However, if you expect to work with large datasets or run multiple instances of Prolog simultaneously, consider upgrading to 8GB or more.
3. **Storage:** Since Prolog programs and databases typically don't occupy much disk space, a standard hard drive or solid-state drive (SSD) with at least 128GB of storage should be sufficient. Consider additional storage if you anticipate storing large datasets locally.
4. **Operating System:** The diet planner system should be compatible with various operating systems, including Windows, macOS, and Linux. Ensure that your hardware supports the operating system of your choice.
5. **Graphics:** Basic integrated graphics are adequate for running Prolog applications. Dedicated graphics cards are unnecessary unless you're performing intensive graphical tasks alongside the diet planner.
6. **Network:** A stable internet connection is recommended if the diet planner relies on online resources or cloud-based services for data retrieval or updates. However, the diet planner itself doesn't require high network bandwidth or low latency.
7. **Peripheral Devices:** Standard input devices such as a keyboard and mouse are necessary for interacting with the diet planner. Additionally, a display monitor with a resolution of at least 1280x800 pixels is recommended for comfortable viewing.
8. **Backup:** Implement a backup solution to prevent data loss in case of hardware failure or accidental deletion. This could involve regular backups to an external storage device or cloud-based backup services.

Overall, the hardware requirements for the diet planner project are modest, focusing more on standard components and ensuring compatibility with the chosen operating system.

## 1.4 Software Specification

The objective of the diet planner software is to provide users with a convenient and personalized tool for meal planning based on their dietary preferences and nutritional requirements. The software aims to promote healthier eating habits and simplify the process of creating balanced meal plans.

1. **Functional Requirements:**

- **Meal Generation:** The software should generate daily meal plans based on user preferences, including vegetarian or non-vegetarian choices, calorie intake goals, and nutritional balance requirements.

**- Nutritional Information:** Detailed nutritional information for each meal should be provided, including calories, protein, carbohydrates, and fats.

**- Food Database:** The software should maintain a comprehensive database of food items categorized by type (e.g., beverage, food, fruit) and nutritional content.

**- User Input:** Users should be able to input their dietary preferences, including likes, dislikes, allergies, and any specific dietary restrictions.

1. **Non-Functional Requirements:**

**- Performance:** The software should be responsive and able to handle user requests simultaneously without significant delay.

**- Scalability:** The software should be scalable to accommodate a growing user base and database of food items.

**- Reliability:** The software should be reliable, with minimal downtime or errors during operation.

**- Compatibility:** The software should be compatible with various devices and platforms, including desktops, tablets, and mobile phones, and support multiple web browsers.

**- Documentation:** Comprehensive documentation should be provided for users and developers, including user guides, API documentation, and technical specifications.

1. **Technology Stack:**

**- Programming Language:** Prolog for backend logic and database management.

**- Database:** Prolog database for storing food items and their nutritional information.

**- Web Framework:** No specific web framework required, as Prolog will handle backend logic.

1. **Constraints:**

**- Data Availability:** The accuracy and completeness of meal plans depend on the availability and quality of data in the food database.

**- User Input:** The effectiveness of meal plans may vary based on the accuracy and specificity of user input regarding dietary preferences and requirements.

1. **Assumptions:**

**- Nutritional Accuracy:** Nutritional information provided for food items in the database is assumed to be accurate and up-to-date.

**- User Understanding:** Users are assumed to have a basic understanding of nutrition and dietary requirements when inputting preferences and selecting meal options.

# Chapter 2: Background and Alternatives

## 2.1 Prolog

Prolog is a logic programming language that is particularly well-suited for developing systems that deal with symbolic information and rule-based reasoning. Developed in the early 1970s, Prolog is based on formal logic, specifically the Horn clause subset of first-order logic. It distinguishes itself from procedural languages like C or Java by focusing on declarative programming, where programmers specify what they want to achieve rather than how to achieve it.

The central idea in Prolog is the definition of relations and rules. Programs consist of a collection of facts (statements about the world) and rules (logical implications). These facts and rules are used to query the knowledge base, and Prolog's inference engine determines whether a given query is provable based on the provided knowledge.

Prolog's pattern matching and backtracking mechanisms are crucial for its operation. Pattern matching allows Prolog to unify query terms with facts and rules in the knowledge base, while backtracking enables exploration of alternative solutions when a query fails.

In the context of the project, Prolog serves as the backbone for developing a diet planner system. Its logical and rule-based nature makes it well-suited for representing dietary rules and preferences, enabling the generation of personalized meal plans based on user inputs.

## 2.2 Diet Planner

The diet planner component of the project is responsible for generating personalized meal plans based on user preferences and dietary requirements. It encompasses several key functionalities:

1. **Food Database:** A comprehensive database of food items containing information such as name, category (e.g., fruits, vegetables, grains), nutritional content (e.g., calories, protein, carbohydrates, fats), and suitability for different meal times (e.g., breakfast, lunch, dinner).
2. **Personalization:** The diet planner takes into account user-specific factors such as dietary preferences (e.g., vegetarian, vegan, omnivore), dietary restrictions (e.g., allergies, intolerances), calorie intake goals, and nutritional balance requirements.
3. **Meal Generation:** Leveraging Prolog's rule-based approach, the system generates daily meal plans by selecting appropriate food items from the database based on user preferences and nutritional criteria. Rules can be defined to ensure that each meal meets specific nutritional goals (e.g., minimum protein intake, maximum calorie limit).
4. **Nutritional Information:** The diet planner provides detailed nutritional information for each meal, including total calorie count and macronutrient breakdown (protein, carbohydrates, fats). This information enables users to make informed decisions about their dietary choices and track their nutritional intake over time.

## 2.3 Types

The project incorporates various types of food items, each offering distinct nutritional profiles and mealtime suitability. These types include:

1. **Beverages:** This category includes drinks such as water, juices, teas, and coffee. Beverages provide hydration and can contribute to daily fluid intake but often contain minimal calories and nutrients.
2. **Fruits:** Fruits are rich sources of vitamins, minerals, fiber, and antioxidants. They are typically low in calories and fat, making them ideal for snacking or as components of meals and desserts.
3. **Vegetarian Meals:** Vegetarian meals consist of plant-based foods such as fruits, vegetables, grains, legumes, nuts, seeds, and dairy products. Vegetarian diets are associated with various health benefits, including reduced risk of heart disease, hypertension, type 2 diabetes, and certain cancers.
4. **Non-vegetarian Meals:** Non-vegetarian meals contain animal-derived foods such as meat, poultry, fish, eggs, and dairy products. These foods are rich sources of high-quality protein, essential amino acids, vitamins (e.g., B12, D), and minerals (e.g., iron, zinc). Non-vegetarian diets can provide important nutrients but may also be associated with environmental and ethical considerations.

## 2.4 Alternatives

While Prolog offers a powerful platform for developing the diet planner, alternative approaches and technologies exist for implementing similar systems:

1. **Machine Learning:** Machine learning techniques, such as natural language processing (NLP) and recommendation systems, can be used to analyze dietary patterns, predict meal preferences, and generate personalized meal plans. Deep learning models, such as recurrent neural networks (RNNs) and transformer-based architectures (e.g., GPT), can learn complex patterns from dietary data and provide personalized recommendations.
2. **Web-Based Applications:** Web-based applications developed using frameworks like Django, Flask, or Node.js can provide interactive and user-friendly interfaces for meal planning and nutrition tracking. These applications can leverage client-side scripting languages (e.g., JavaScript) and web APIs to create dynamic and responsive user experiences.
3. **Mobile Applications:** Mobile applications for iOS and Android platforms, built using frameworks like React Native or Flutter, can offer on-the-go access to meal planning functionalities. Mobile apps can utilize device features such as camera integration for food recognition, barcode scanning for product identification, and geolocation for local food recommendations.
4. **Nutrition APIs:** Integration with nutrition APIs such as the USDA FoodData Central API or the Nutritionix API can provide access to extensive databases of food items and nutritional information. These APIs offer standardized data formats (e.g., JSON, XML) and endpoints for querying food details, enabling the development of diet planning features without manually curating a food database.

Each alternative approach presents unique advantages and challenges, and the choice depends on factors such as project requirements, technical expertise, and user preferences. However, the Prolog-based implementation offers a robust and flexible solution with the potential for sophisticated rule-based reasoning and customization capabilities.

# Chapter 3: Problem Formulation

The focal point of this chapter is the intricate examination of the challenges encountered by individuals in managing their dietary habits, particularly amidst the complexities of modern lifestyles and diverse dietary preferences. The overarching issue revolves around the lack of accessible and efficient tools to facilitate effective meal planning, leading to suboptimal nutrition, time constraints, and a general lack of awareness regarding dietary needs. The crux of the matter lies in the absence of a comprehensive platform that integrates personalized meal planning, nutritional guidance, and educational resources to empower individuals in making informed dietary choices.

**3.1 Key elements of the issue:**

**i. Complexity of Meal Planning:**

Meal planning entails a multitude of considerations, including nutritional balance, dietary preferences, health goals, and time constraints. Many individuals find it challenging to navigate these complexities effectively, resulting in reliance on convenient but often unhealthy food options and suboptimal dietary habits.

**ii. Diverse Dietary Preferences:**

Individuals have diverse dietary preferences and restrictions, including vegetarian, vegan, gluten-free, and various cultural or health-related considerations. Accommodating these preferences while ensuring adequate nutrition and variety adds complexity to the meal planning process.

**iii. Nutritional Awareness:**

There is a widespread lack of awareness regarding the nutritional content of different foods and how to balance one's diet effectively. This leads to poor dietary choices, nutrient imbalances, and potential health issues such as obesity, malnutrition, or nutrient deficiencies.

**iv. Time Constraints:**

Modern lifestyles often leave individuals with limited time for meal preparation and planning. This results in a reliance on fast food or pre-packaged meals, which may be convenient but are often lacking in nutritional quality. Time constraints further exacerbate the challenges of maintaining a healthy diet.

**3.2 Proposed Solution:**

The proposed solution entails the development of a comprehensive diet planner system aimed at addressing the aforementioned challenges. This system will leverage artificial intelligence (AI) and database technologies to provide personalized meal plans, nutritional guidance, and educational resources to users. Key elements of the proposed solution include:

**i. Database of Food Items:**

Curating a database of diverse food items categorized by type, nutritional content, and suitability for different dietary preferences and restrictions.

**ii. User Profiles:**

Allowing users to create personalized profiles with information such as dietary preferences, health goals, allergies, and cultural considerations.

**iii. Algorithmic Meal Planning:**

Developing algorithms that generate personalized meal plans based on user profiles, nutritional guidelines, and dietary preferences. These plans will ensure nutritional balance, variety, and adherence to individual dietary needs.

**iv. Nutritional Guidance:**

Providing detailed nutritional information for each meal, including calorie count, macronutrient distribution, and micronutrient content. This will empower users to make informed decisions about their diet and track their nutritional intake.

**v. User-Friendly Interface:**

Designing an intuitive and user-friendly interface accessible via web or mobile platforms. The interface will allow users to input their preferences, view meal plans, customize recipes, and access educational resources.

**vi. Educational Resources:**

Offering articles, recipes, and tips on healthy eating, meal preparation, and nutritional awareness. These resources will empower users to develop sustainable dietary habits and make healthier food choices.

In essence, the proposed diet planner system aims to simplify meal planning, promote healthier eating habits, and empower individuals to take control of their nutrition and overall well-being. The subsequent chapters will delve into the methodologies employed in the development of this innovative system, the experimental setup, and the anticipated outcomes.

# Chapter 4: Objectives

The objectives of the "Diet Planner" project are meticulously crafted to address the multifaceted challenges faced by individuals in maintaining their healthy lifestyle. Each objective is designed to contribute to the overarching goal of creating an inclusive and transformative platform.

1. **Develop a Comprehensive Food Database:** Create a detailed database of food items encompassing various categories such as beverages, snacks, meals, fruits, and vegetables. Each entry in the database should include essential nutritional information such as calories, protein, carbohydrates, and fats.
2. **Implement Meal Categorization:** Classify food items into appropriate meal categories based on their suitability for different times of the day, including early morning, breakfast, snack, lunch, evening snack, and dinner. Ensure that the categorization takes into account factors like energy content, and nutritional balance.
3. **Personalize Meal Plans:** Utilize Prolog's logic programming capabilities to generate personalized meal plans for users based on their individual preferences and nutritional requirements. Consider factors like calorie intake, protein intake, dietary restrictions (e.g., vegetarian or non-vegetarian), and mealtime preferences.
4. **Ensure Accuracy and Reliability:** Implement rigorous testing procedures to validate the accuracy and reliability of the meal planning algorithms. Conduct extensive testing with diverse user inputs and scenarios to identify and rectify any discrepancies or inaccuracies in the generated meal plans.
5. **Enhance Accessibility:** Ensure that the diet planner system is accessible to a wide range of users, including those with visual impairments or disabilities. Implement accessibility features such as screen reader compatibility, keyboard navigation support, and high contrast themes to enhance usability for all users.
6. **Foster Inclusivity:** Accommodate diverse dietary preferences and restrictions within the meal planning system. Include options for vegetarian, non-vegetarian, vegan, gluten-free, and other specialized diets to cater to the needs of a diverse user base.
7. **Provide Nutritional Information:** Display detailed nutritional information for each food item and meal plan generated by the system. Present information such as calorie count, protein content, carbohydrate content, fat content, vitamins, and minerals to help users make informed decisions about their diet.
8. **Educate Users:** Offer educational resources and tips within the diet planner system to promote awareness of healthy eating habits and nutrition. Provide guidance on portion control, balanced meals, and the importance of incorporating fruits, vegetables, and whole grains into the diet.
9. **Continuously Improve and Update:** Gather feedback from users and iterate on the diet planner system based on user suggestions and evolving dietary guidelines. Regularly update the food database with new entries and ensure that the system remains up-to-date with the latest nutritional information and dietary recommendations.

# Chapter 5: Methodology

The methodology for the "Diet Planner" project is structured to ensure the successful development of an accessible, user-friendly, and feature-rich solution.

**5.1 Phases**

The following phases delineate the systematic approach that will be adhered to:

**i. Project Initiation:**

**Needs Assessment:** Before embarking on the project, a thorough needs assessment is conducted. This involves identifying specific requirements and gaining insights into the preferences of individuals. Understanding user needs is foundational to designing a diet planner that truly caters to the target audience.

**Project Planning:** With insights from the needs assessment, a comprehensive project plan is developed. This plan outlines clear objectives, establishes deadlines, allocates resources effectively, and outlines financial requirements. Project planning sets the stage for a structured and organized development process.

**ii. Research and Requirements Gathering:**

- Thorough research was conducted to understand the needs and preferences of potential users regarding meal planning.

- Key features and functionalities required in the diet planner system were identified based on the research findings.

- User personas were defined to represent various user demographics and dietary preferences.

**iii. Database Creation:**

- A comprehensive database of food items was created, categorizing them based on type (beverage, food, fruit, etc.), mealtime suitability, and nutritional content (calories, protein, carbs, fat).

- The database was populated with accurate and up-to-date nutritional information sourced from reliable sources such as nutrition databases and food labeling.

**iv. System Design:**

- The architecture of the diet planner system was designed, considering scalability, modularity, and ease of maintenance.

- Data structures and relationships necessary to efficiently store and retrieve information from the food database were defined.

- User interface components and layout were determined to ensure an intuitive and user-friendly experience.

**v. Implementation:**

- The diet planner system was developed using Prolog programming language, leveraging its logic-based approach to represent and manipulate data.

- Predicates were implemented to categorize foods, generate meal plans, and calculate nutritional values based on user input.

- User interface components were designed and integrated for interacting with the system, ensuring seamless navigation and accessibility.

**vi. Testing and Validation:**

- Extensive testing was conducted to validate the functionality and accuracy of the diet planner system.

- Unit tests were performed to verify the correctness of individual predicates and modules.

- User acceptance testing was conducted with a diverse group of users to gather feedback and identify areas for improvement.

**vii. Refinement and Optimization:**

- Iterations were made on the design and implementation based on user feedback and testing results.

- The performance of the system was optimized, focusing on speed and efficiency in generating meal plans and retrieving nutritional information.

- Usability issues or bugs identified during testing were addressed, ensuring a smooth and seamless user experience.

**viii. Documentation:**

- The design, implementation, and functionality of the diet planner system were documented in detail.

- User manuals and guides were provided to help users navigate and utilize the system effectively.

- Assumptions, constraints, or limitations associated with the system were documented.

**ix.Deployment and Maintenance:**

- The diet planner system was deployed on a suitable platform or server for users to access.

- The performance and usage of the system were monitored, and any issues or errors that arose were addressed.

- Ongoing maintenance and support were provided, including updates to the food database and system functionality as needed.

This detailed methodology provides a step-by-step guide to the development of the "Diet Planner" Each phase is carefully crafted to contribute to the overarching goal of creating an inclusive, user-friendly, and feature-rich digital solution for individuals.

**5.2 Timeline**

|  |  |  |
| --- | --- | --- |
| Number of Task | Task Name | Required time in days |
| 1 | Proposal | 1 |
| 2 | Information planning and collecting | 3 |
| 3 | Background and alternatives | 1 |
| 4 | Determine system requirements | 1 |
| 5 | System requirement analysis and design | 2 |
| 6 | System programming and configuration | 4 |
| 7 | System Test | 3 |
| 8 | System Documentation | All project period |

Table 5.1 Timeline

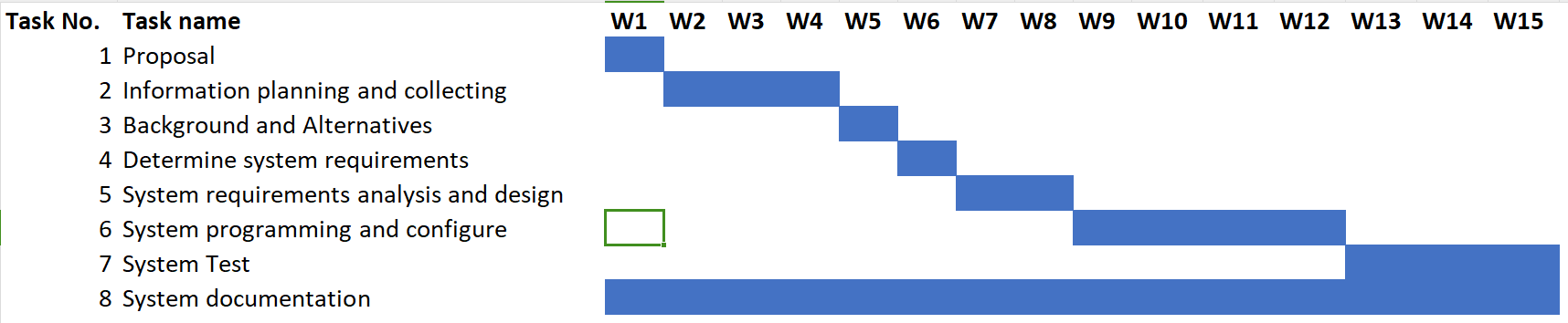


Fig 5.1 Timeline Gantt Chart

**5.3 Use Case:**In the methodology, the use case described above represents the interaction between the user and the diet planner system. Here's an explanation of each step within the use case:

**i. Enter User Details:** The user (U) initiates the interaction by entering their personal details, such as age, gender, weight, height, activity level, and dietary restrictions into the system. This information is essential for generating a personalized diet plan.

**ii. Generate Diet Plan:** After entering the user details, the system processes this information and generates a customized diet plan tailored to the user's specific requirements and goals. This plan includes recommended meals and portion sizes for each mealtime, taking into account the user's nutritional needs and preferences.

**iii. Show Nutritional Values:** Once the diet plan is generated, the user can view the nutritional values associated with each meal, including calorie count, macronutrient distribution (protein, carbohydrates, fats), and micronutrient content (vitamins, minerals). This allows the user to understand the nutritional composition of their meals and make informed decisions about their diet.

**iv. Interaction Flow:** The user interacts with the system in a sequential manner, starting with entering their details, then generating the diet plan, and finally viewing the nutritional values. Each step in the process is connected, with the input provided in one step influencing the output in subsequent steps. The system performs internal processes, such as generating nutritional values based on the diet plan, which are hidden from the user but essential for the functionality of the system. The interaction flow ensures a smooth and intuitive user experience, guiding the user through the necessary steps to obtain a personalized diet plan and nutritional information.

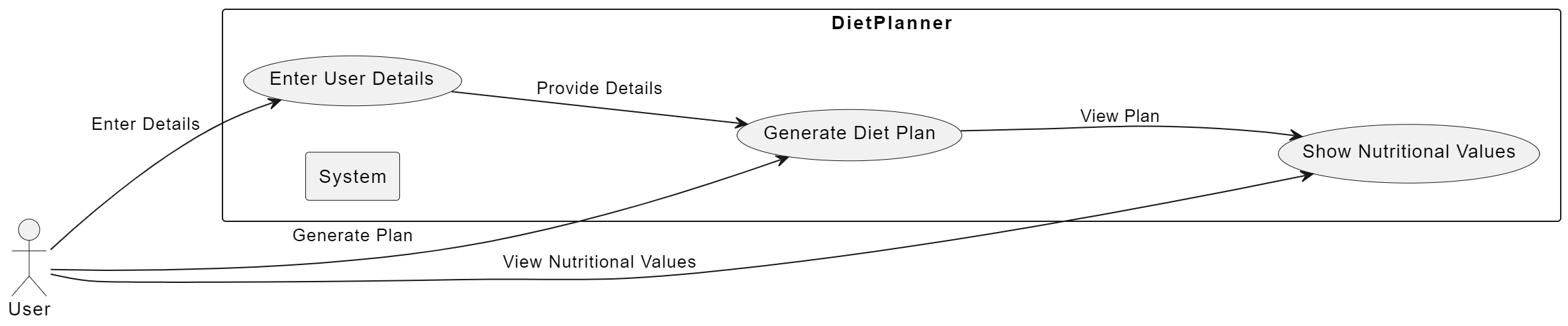


Figure 5.2 Use Case Diagram

**5.4 Code:**% facts about available foods, type, veg/non\_veg, time, calories, protein, carbs and fat values

% early morning food for both

food(cinnamon\_lemon\_water, beverage, veg, early\_morning, 0, 0, 0, 0).

food(zeera\_water, beverage, veg, early\_morning, 0, 0, 0, 0).

food(coconut\_water, beverage, veg, early\_morning, 45, 0.5, 11, 0).

food(saunf\_water, beverage, veg, early\_morning, 0, 0, 0, 0).

food(chia\_seed\_water, beverage, veg, early\_morning, 138, 4.4, 12.3, 8.6).

food(lemon\_honey\_water, beverage, veg, early\_morning, 94, 0.2, 25.6, 0).

food(apple\_cider\_vinegar\_water, early\_morning, beverage, veg, 22, 0, 0.9, 0).

food(ginger\_water, beverage, veg, early\_morning, 0, 0, 0, 0).

food(fenugreek\_water, beverage, veg, early\_morning, 0, 0, 0, 0).

food(aloe\_vera\_juice, beverage, veg, early\_morning, 0, 0, 0, 0).

food(black\_coffee, beverage, veg, early\_morning, 2, 0.3, 0, 0).

food(green\_tea, beverage, veg, early\_morning, 0, 0, 0, 0).

% breakfast\_food for veg

food(upma, food, veg, breakfast, 250, 6, 45, 5).

food(oats, food, veg, breakfast, 150, 5, 27, 3).

food(daliya, food, veg, breakfast, 200, 6, 40, 3).

food(poha, food, veg, breakfast, 250, 5, 45, 5).

food(namkeen\_seviyan, food, veg, breakfast, 200, 5, 40, 5).

food(salad, food, veg, breakfast, 100, 2, 20, 1).

food(banana\_smoothie, food, veg, breakfast, 150, 3, 30, 2).

food(chilla, food, veg, breakfast, 300, 15, 30, 10).

food(veg\_sandwich, food, veg, breakfast, 300, 15, 30, 10).

food(veg\_salad, food, veg, breakfast, 300, 15, 30, 10).

food(veg\_soup, food, veg, breakfast, 300, 15, 30, 10).

% breakfast\_food for non-veg

food(egg\_white\_omlete, food, non\_veg, breakfast, 200, 10, 20, 10).

food(oats, food, non\_veg, breakfast, 150, 5, 27, 3).

food(daliya, food, non\_veg, breakfast, 200, 6, 40, 3).

food(poha, food, non\_veg, breakfast, 250, 5, 45, 5).

food(paneer\_sandwich, food, non\_veg, breakfast, 300, 15, 30, 10).

food(salad, food, non\_veg, breakfast, 100, 2, 20, 1).

food(banana\_smoothie, food, non\_veg, breakfast, 150, 3, 30, 2).

food(peanut\_butter\_sandwich, food, non\_veg, breakfast, 300, 15, 30, 10).

food(chicken\_sandwich, food, non\_veg, breakfast, 300, 15, 30, 10).

food(chicken\_salad, food, non\_veg, breakfast, 300, 15, 30, 10).

food(chicken\_soup, food, non\_veg, breakfast, 300, 15, 30, 10).

% snack for veg

food(apple, fruit, veg, snack, 100, 1, 20, 0).

food(banana, fruit, veg, snack, 100, 1, 20, 0).

food(grapes, fruit, veg, snack, 100, 1, 20, 0).

food(orange, fruit, veg, snack, 100, 1, 20, 0).

food(pomegranate, fruit, veg, snack, 100, 1, 20, 0).

food(watermelon, fruit, veg, snack, 100, 1, 20, 0).

food(muskmelon, fruit, veg, snack, 100, 1, 20, 0).

food(yogurt, food, veg, snack, 100, 5, 10, 5).

food(pistas, nuts, veg, snack, 100, 5, 10, 5).

food(coconut\_water, beverage, veg, snack, 45, 0.5, 11, 0).

food(vegetable\_juice, beverage, veg, snack, 100, 1, 20, 0).

food(buttermilk, beverage, veg, snack, 100, 5, 10, 5).

% snack for non-veg

food(apple, fruit, non\_veg, snack, 100, 1, 20, 0).

food(banana, fruit, non\_veg, snack, 100, 1, 20, 0).

food(grapes, fruit, non\_veg, snack, 100, 1, 20, 0).

food(orange, fruit, non\_veg, snack, 100, 1, 20, 0).

food(pomegranate, fruit, non\_veg, snack, 100, 1, 20, 0).

food(watermelon, fruit, non\_veg, snack, 100, 1, 20, 0).

food(muskmelon, fruit, non\_veg, snack, 100, 1, 20, 0).

food(yogurt, food, non\_veg, snack, 100, 5, 10, 5).

food(pistas, nuts, non\_veg, snack, 100, 5, 10, 5).

food(coconut\_water, beverage, non\_veg, snack, 45, 0.5, 11, 0).

food(vegetable\_juice, beverage, non\_veg, snack ,100, 1, 20, 0).

food(buttermilk, beverage, non\_veg, snack, 100, 5, 10, 5).

% lunch for veg

food(grilled\_paneer\_salad, food, veg, lunch, 350, 18, 25, 20).

food(sprout\_rice, food, veg, lunch, 400, 12, 30, 15).

food(fried\_rice, food, veg, lunch, 450, 10, 40, 22).

food(veg\_kebabs, food, veg, lunch, 300, 15, 20, 18).

food(rajma\_rice, food, veg, lunch, 380, 14, 35, 16).

food(dal\_khichdi, food, veg, lunch, 320, 12, 28, 14).

food(sambar\_rice, food, veg, lunch, 360, 10, 30, 20).

food(roti\_sabzi, food, veg, lunch, 320, 10, 35, 15).

food(vegetable\_soup, food, veg, lunch, 150, 5, 20, 7).

food(veg\_pulao, food, veg, lunch, 400, 10, 35, 18).

food(veg\_biryani, food, veg, lunch, 450, 12, 40, 20).

food(sambar\_idli, food, veg, lunch, 300, 8, 25, 12).

% lunch for non-veg

food(grilled\_chicken\_salad, food, non\_veg, lunch, 250, 25, 10, 12).

food(chicken\_rice, food, non\_veg, lunch, 350, 30, 20, 14).

food(steamed\_chicken\_balls, food, non\_veg, lunch, 300, 28, 15, 10).

food(rice\_with\_fish\_curry, food, non\_veg, lunch, 400, 22, 25, 16).

food(chicken\_biryani, food, non\_veg, lunch, 450, 35, 30, 18).

food(chicken\_kebabs, food, non\_veg, lunch, 280, 27, 10, 15).

food(chicken\_soup, food, non\_veg, lunch, 200, 18, 8, 10).

food(chicken\_pulao, food, non\_veg, lunch, 380, 33, 22, 14).

food(chicken\_fried\_rice, food, non\_veg, lunch, 320, 28, 18, 12).

food(chicken\_curry, food, non\_veg, lunch, 300, 26, 20, 10).

food(chicken\_biryani, food, non\_veg, lunch, 420, 36, 28, 16).

food(chicken\_idli, food, non\_veg, lunch, 280, 24, 15, 10).

% evening snack for veg

food(oats\_soup, food, veg, evening\_snack, 70, 1.8, 13.2, 1.4).

food(carrot\_sticks, vegetable, veg, evening\_snack, 30, 0.6, 7, 0.1).

food(mango\_juice, beverage, veg, evening\_snack, 120, 0.8, 30, 0.4).

food(paneer\_chat, food, veg, evening\_snack, 150, 6, 12, 9).

food(roasted\_makhana, food, veg, evening\_snack, 50, 2, 8, 1).

food(roasted\_chana, food, veg, evening\_snack, 90, 5, 15, 2).

food(coconut\_water, beverage, veg, evening\_snack, 45, 0.5, 11, 0).

food(vegetable\_juice, beverage, veg, evening\_snack, 50, 1, 12, 0.5).

food(apple\_juice, beverage, veg, evening\_snack, 60, 0.2, 15, 0).

food(banana\_juice, beverage, veg, evening\_snack, 80, 1, 20, 0.5).

food(alovera\_juice, beverage, veg, evening\_snack, 70, 0.2, 17, 0).

food(mix\_fruit\_juice, beverage, veg, evening\_snack, 100, 0.5, 25, 0.5).

% evening snack for non-veg

food(fish\_soup, food, non\_veg, evening\_snack, 120, 8, 3, 7).

food(chicken\_sticks, food, non\_veg, evening\_snack, 250, 18, 12, 13).

food(mango\_juice, beverage, non\_veg, evening\_snack, 120, 0.5, 30, 0).

food(makhana\_paneer\_chat, food, non\_veg, evening\_snack, 180, 9, 15, 10).

food(potato\_spinach\_omelette, food, non\_veg, evening\_snack, 200, 11, 15, 12).

food(egg\_avocado, food, non\_veg, evening\_snack, 300, 14, 11, 20).

food(egg\_toast, food, non\_veg, evening\_snack, 180, 9, 18, 8).

food(chicken\_nuggets, food, non\_veg, evening\_snack, 280, 20, 15, 16).

food(chicken\_popcorn, food, non\_veg, evening\_snack, 300, 22, 13, 18).

food(chicken\_soup, food, non\_veg, evening\_snack, 150, 10, 5, 9).

food(chicken\_salad, food, non\_veg, evening\_snack, 200, 18, 10, 8).

food(chicken\_sandwich, food, non\_veg, evening\_snack, 350, 25, 20, 15).

% dinner for veg

food(veg\_paneer, food, veg, dinner, 200, 18, 5, 12).

food(soup\_and\_grilled\_paneer, food, veg, dinner, 300, 25, 8, 15).

food(stir\_fry\_veggies, food, veg, dinner, 150, 4, 12, 9).

food(scrambled\_paneer, food, veg, dinner, 250, 20, 4, 16).

food(chapati\_and\_seasonal\_sabji, food, veg, dinner, 300, 8, 25, 12).

food(soya\_roll, food, veg, dinner, 200, 12, 15, 10).

food(cheese\_roll, food, veg, dinner, 250, 15, 18, 12).

food(paneer\_roll, food, veg, dinner, 300, 20, 22, 14).

food(veg\_roll, food, veg, dinner, 200, 10, 20, 8).

food(uttapam, food, veg, dinner, 250, 6, 20, 16).

food(dosa, food, veg, dinner, 200, 5, 25, 10).

% dinner for non-veg

food(chicken\_veggie\_rice, meal, non\_veg, dinner, 300, 25, 30, 10).

food(chicken\_soup, meal, non\_veg, dinner, 250, 20, 15, 8).

food(grilled\_chicken\_broccoli, meal, non\_veg, dinner, 350, 30, 10, 15).

food(egg\_wrap, meal, non\_veg, dinner, 200, 15, 20, 8).

food(chicken\_wrap, meal, non\_veg, dinner, 300, 25, 25, 12).

food(steamed\_fish, meal, non\_veg, dinner, 250, 25, 5, 12).

food(chicken\_breast\_brown\_rice, meal, non\_veg, dinner, 400, 35, 40, 15).

food(chicken\_fry, meal, non\_veg, dinner, 350, 30, 20, 18).

food(chicken\_curry, meal, non\_veg, dinner, 350, 30, 25, 15).

food(fish\_curry, meal, non\_veg, dinner, 300, 25, 20, 15).

food(chicken\_biryani, meal, non\_veg, dinner, 450, 35, 50, 20).

food(fish\_biryani, meal, non\_veg, dinner, 400, 30, 45, 18).

food(fish\_soup, meal, non\_veg, dinner, 250, 20, 15, 8).

% Predicates for each meal category

early\_morning\_food(Food) :- food(Food, \_, veg, early\_morning, \_, \_, \_, \_).

breakfast\_food\_veg(Food) :- food(Food, \_, veg, breakfast, \_, \_, \_, \_).

breakfast\_food\_non\_veg(Food) :- food(Food, \_, non\_veg, breakfast, \_, \_, \_, \_).

snack\_food\_veg(Food) :- food(Food, \_, veg, snack, \_, \_, \_, \_).

snack\_food\_non\_veg(Food) :- food(Food, \_, non\_veg, snack, \_, \_, \_, \_).

lunch\_food\_veg(Food) :- food(Food, \_, veg, lunch, \_, \_, \_, \_).

lunch\_food\_non\_veg(Food) :- food(Food, \_, non\_veg, lunch, \_, \_, \_, \_).

evening\_snack\_food\_veg(Food) :- food(Food, \_, veg, evening\_snack, \_, \_, \_, \_).

evening\_snack\_food\_non\_veg(Food) :- food(Food, \_, non\_veg, evening\_snack, \_, \_, \_, \_).

dinner\_food\_veg(Food) :- food(Food, \_, veg, dinner, \_, \_, \_, \_).

dinner\_food\_non\_veg(Food) :- food(Food, \_, non\_veg, dinner, \_, \_, \_, \_).

% random\_member/2 predicate

random\_member(X, List) :-

length(List, Length),

random(0, Length, Index),

nth0(Index, List, X).

% meal plans for vegetarian

generate\_meal\_plan\_veg(Early\_Morning, Breakfast, Snack, Lunch, Evening\_Snack, Dinner) :-

generate\_early\_morning(Early\_Morning),

generate\_breakfast\_veg(Breakfast),

generate\_snack\_veg(Snack),

generate\_lunch\_veg(Lunch),

generate\_evening\_snack\_veg(Evening\_Snack),

generate\_dinner\_veg(Dinner).

% early morning meal

generate\_early\_morning(Meal) :-

findall(Food, (food(Food, \_, veg, early\_morning, \_, \_, \_, \_), early\_morning\_food(Food)), Early\_MorningOptionsVeg),

random\_member(Meal, Early\_MorningOptionsVeg).

% breakfast meal

generate\_breakfast\_veg(Meal) :-

findall(Food, (food(Food, \_, veg, breakfast, \_, \_, \_, \_), breakfast\_food\_veg(Food)), BreakfastOptionsVeg),

random\_member(Meal, BreakfastOptionsVeg).

% snack meal

generate\_snack\_veg(Meal) :-

findall(Food, (food(Food, \_, veg, snack, \_, \_, \_, \_), snack\_food\_veg(Food)), SnackOptionsVeg),

random\_member(Meal, SnackOptionsVeg).

% lunch meal

generate\_lunch\_veg(Meal) :-

findall(Food, (food(Food, \_, veg, lunch, \_, \_, \_, \_), lunch\_food\_veg(Food)), LunchOptionsVeg),

random\_member(Meal, LunchOptionsVeg).

% evening snack meal

generate\_evening\_snack\_veg(Meal) :-

findall(Food, (food(Food, \_, veg, evening\_snack, \_, \_, \_, \_), evening\_snack\_food\_veg(Food)), Evening\_SnackOptionsVeg),

random\_member(Meal, Evening\_SnackOptionsVeg).

% dinner meal

generate\_dinner\_veg(Meal) :-

findall(Food, (food(Food, \_, veg, dinner, \_, \_, \_, \_), dinner\_food\_veg(Food)), DinnerOptionsVeg),

random\_member(Meal, DinnerOptionsVeg).% Define predicates for each day of the week

% Define rules to generate meal plans for vegetarian

generate\_meal\_plan\_non\_veg(Early\_Morning, Breakfast, Snack, Lunch, Evening\_Snack, Dinner) :-

generate\_early\_morning(Early\_Morning),

generate\_breakfast\_non\_veg(Breakfast),

generate\_snack\_non\_veg(Snack),

generate\_lunch\_non\_veg(Lunch),

generate\_evening\_snack\_non\_veg(Evening\_Snack),

generate\_dinner\_non\_veg(Dinner).

% early morning meal

generate\_early\_morning(Meal) :-

findall(Food, (food(Food, \_, veg, early\_morning, \_, \_, \_, \_), early\_morning\_food(Food)), Early\_MorningOptionsNonVeg),

random\_member(Meal, Early\_MorningOptionsNonVeg).

% breakfast meal

generate\_breakfast\_non\_veg(Meal) :-

findall(Food, (food(Food, \_, non\_veg, breakfast, \_, \_, \_, \_), breakfast\_food\_non\_veg(Food)), BreakfastOptionsNonVeg),

random\_member(Meal, BreakfastOptionsNonVeg).

% snack meal

generate\_snack\_non\_veg(Meal) :-

findall(Food, (food(Food, \_, non\_veg, snack, \_, \_, \_, \_), snack\_food\_non\_veg(Food)), SnackOptionsNonVeg),

random\_member(Meal, SnackOptionsNonVeg).

% lunch meal

generate\_lunch\_non\_veg(Meal) :-

findall(Food, (food(Food, \_, non\_veg, lunch, \_, \_, \_, \_), lunch\_food\_non\_veg(Food)), LunchOptionsNonVeg),

random\_member(Meal, LunchOptionsNonVeg).

% evening snack meal

generate\_evening\_snack\_non\_veg(Meal) :-

findall(Food, (food(Food, \_, non\_veg, evening\_snack, \_, \_, \_, \_), evening\_snack\_food\_non\_veg(Food)), Evening\_SnackOptionsNonVeg),

random\_member(Meal, Evening\_SnackOptionsNonVeg).

% dinner meal

generate\_dinner\_non\_veg(Meal) :-

findall(Food, (food(Food, \_, non\_veg, dinner, \_, \_, \_, \_), dinner\_food\_non\_veg(Food)), DinnerOptionsNonVeg),

random\_member(Meal, DinnerOptionsNonVeg).% Define predicates for each day of the week

% Predicates for each day of the week

day(monday).

day(tuesday).

day(wednesday).

day(thursday).

day(friday).

day(saturday).

day(sunday).

% Predicate to generate veg meals for each day

generate\_daily\_meals\_veg(Day, Meals) :-

generate\_early\_morning(EarlyMorning),

generate\_breakfast\_veg(Breakfast),

generate\_snack\_veg(Snack),

generate\_lunch\_veg(Lunch),

generate\_evening\_snack\_veg(EveningSnack),

generate\_dinner\_veg(Dinner),

Meals = [EarlyMorning, Breakfast, Snack, Lunch, EveningSnack, Dinner].

% Predicate to generate meals for each day

generate\_daily\_meals\_non\_veg(Day, Meals) :-

generate\_early\_morning(EarlyMorning),

generate\_breakfast\_non\_veg(Breakfast),

generate\_snack\_non\_veg(Snack),

generate\_lunch\_non\_veg(Lunch),

generate\_evening\_snack\_non\_veg(EveningSnack),

generate\_dinner\_non\_veg(Dinner),

Meals = [EarlyMorning, Breakfast, Snack, Lunch, EveningSnack, Dinner].

% Predicate to calculate the total nutrition count for a list of meals

total\_nutrition\_count(Meals, TotalCalories, TotalProtein, TotalCarbs, TotalFats) :-

total\_nutrition\_count(Meals, 0.0, 0.0, 0.0, 0.0, TotalCalories, TotalProtein, TotalCarbs, TotalFats).

total\_nutrition\_count([], Calories, Protein, Carbs, Fats, TotalCalories, TotalProtein, TotalCarbs, TotalFats) :-

% Round off the final totals

TotalCaloriesRounded is round(Calories),

TotalProteinRounded is round(Protein),

TotalCarbsRounded is round(Carbs),

TotalFatsRounded is round(Fats),

% Unify the rounded totals

TotalCalories is TotalCaloriesRounded,

TotalProtein is TotalProteinRounded,

TotalCarbs is TotalCarbsRounded,

TotalFats is TotalFatsRounded.

total\_nutrition\_count([Meal|Rest], CaloriesSoFar, ProteinSoFar, CarbsSoFar, FatsSoFar, TotalCalories, TotalProtein, TotalCarbs, TotalFats) :-

food(Meal, \_, \_, \_, Calories, Protein, Carbs, Fats),

% Calculate the new totals

NewCalories is CaloriesSoFar + Calories,

NewProtein is ProteinSoFar + Protein,

NewCarbs is CarbsSoFar + Carbs,

NewFats is FatsSoFar + Fats,

total\_nutrition\_count(Rest, NewCalories, NewProtein, NewCarbs, NewFats, TotalCalories, TotalProtein, TotalCarbs, TotalFats).

print\_daily\_meals(Day, Meals) :-

write(Day), write(': '), nl,

print\_meal\_category(early\_morning, Meals),

print\_meal\_category(breakfast, Meals),

print\_meal\_category(snack, Meals),

print\_meal\_category(lunch, Meals),

print\_meal\_category(evening\_snack, Meals),

print\_meal\_category(dinner, Meals),

nl,

total\_nutrition\_count(Meals, TotalCalories, TotalProtein, TotalCarbs, TotalFats),

write('Total Calories for '), write(Day), write(': '), write(TotalCalories), nl,

write('Total Protein for '), write(Day), write(': '), write(TotalProtein), nl,

write('Total Carbs for '), write(Day), write(': '), write(TotalCarbs), nl,

write('Total Fats for '), write(Day), write(': '), write(TotalFats), nl, nl.

print\_meal\_category(Category, Meals) :-

member(Meal, Meals),

food(Meal, \_, \_, Category, \_, \_, \_, \_),

write(Category), write(': '), write(Meal), nl.

% Print weekly diet schedule for veg

generate\_weekly\_diet\_schedule\_veg :-

% Findall meals for each day and print them directly

findall(Day-Meals, (day(Day), generate\_daily\_meals\_veg(Day, Meals)), VegWeeklySchedule),

print\_schedule(VegWeeklySchedule),

!.

% Print weekly diet schedule for non-veg

generate\_weekly\_diet\_schedule\_non\_veg :-

% Findall meals for each day and print them directly

findall(Day-Meals, (day(Day), generate\_daily\_meals\_non\_veg(Day, Meals)), NonVegWeeklySchedule),

print\_schedule(NonVegWeeklySchedule),

!.

print\_schedule([]).

print\_schedule([Day-Meals|Rest]) :-

print\_daily\_meals(Day, Meals),

print\_schedule(Rest).

% Predicate to calculate BMI

calculate\_bmi(Weight, Height, BMI) :-

HeightMeters is Height / 100,

BMI is Weight / (HeightMeters \* HeightMeters).

% Predicate to categorize BMI

categorize\_bmi(BMI, Category) :-

BMI < 18.5 -> Category = underweight;

BMI >= 18.5, BMI < 25 -> Category = normal\_weight;

Category = overweight.

% Predicate to accept user input

accept\_user\_input :-

write('Enter your name: '),

read(Name),

write('Enter your age: '),

read(Age),

write('Are you vegetarian? (yes/no): '),

read(VegetarianResponse),

(VegetarianResponse == yes ->

Vegetarian = true;

Vegetarian = false

),

write('Enter your weight (in kg): '),

read(Weight),

write('Enter your height (in cm): '),

read(Height),

calculate\_bmi(Weight, Height, BMI),

categorize\_bmi(BMI, Category),

write('Your BMI is: '), write(BMI), nl,

write('You are '), write(Category), nl,

% Determine which diet plan to generate based on users preference

(Vegetarian -> generate\_weekly\_diet\_schedule\_veg; generate\_weekly\_diet\_schedule\_non\_veg).

# Chapter 6: Experimental Setup

The experimental setup for testing and evaluating the "Diet Planner" in Prolog is a foundational step in validating its efficacy in delivering personalized diet plans and nutritional insights. By meticulously designing and executing a comprehensive testing framework, we aim to scrutinize every aspect of the diet planner's functionality and performance. This process is indispensable in ensuring that the system not only meets but exceeds the expectations of its users. Here's an elaboration of the proposed experimental setup:

1. **Test Environment:**

**- Server:** Utilizing a dedicated server or cloud hosting provider like github to deploy the diet planner, ensuring it meets the necessary hardware and software specifications.

**- Development Environment:** Establishing the Prolog development environment, including code repositories (such as Git), and implementing continuous integration/continuous deployment (CI/CD) pipelines for automated testing and deployment in the development environment.

1. **Accessibility Testing:**

**- Validating User Inputs:** Testing the system's ability to handle various user inputs, such as age, gender, weight, height, activity level, and dietary restrictions, ensuring accurate processing and interpretation of data.

**- Data Verification:** Verifying the accuracy and reliability of the data sources used for generating personalized diet plans and nutritional information.

1. **Diet Plan Generation Testing:**

**- Personalized Diet Plan Generation:** Evaluating the system's ability to generate personalized diet plans based on user input and nutritional requirements, ensuring the plans are tailored to individual needs and goals.

**- Nutritional Value Calculation:** Testing the accuracy of nutritional value calculations for each meal in the diet plan, including calorie count, macronutrient distribution, and micronutrient content.

1. **Functional Testing:**

Testing all functionalities of the diet planner, including user registration, meal planning, nutritional information display, and account management, to ensure they operate as intended and meet user expectations.

**- Input Validation Testing:** Validating the diet planner's ability to handle and process user input effectively, including error handling for invalid or incomplete data.

**- Diet Plan Generation Testing:** Assessing the accuracy and relevance of diet plans generated by the system based on user input and dietary goals.

**- Nutritional Information Retrieval Testing:** Verifying the correctness of nutritional information displayed to users, ensuring alignment with established nutritional guidelines and data sources.

1. **User Interface Testing:**

To guarantee that the interface is user-friendly and intuitive for all users, doing UI testing with sighted and visually impaired users.

1. **User Experience Testing:**

**- Usability Testing:** Conducting usability tests with representative users to evaluate the intuitiveness and ease of use of the diet planner's interface and features.

**- Accessibility Testing:** Ensuring that the diet planner interface is accessible to users with disabilities, including those using assistive technologies such as screen readers or voice commands.

1. **Performance Testing:**

**- Load Testing:** Evaluating the performance of the diet planner under different user loads to ensure it can handle concurrent interactions without degradation in performance.

**- Response Time Testing:** Measuring the system's response time for generating diet plans and displaying nutritional information to ensure timely and efficient user interactions.

1. **Monitoring and Maintenance:**

**- Monitoring Tools Implementation:** Deploying monitoring tools to track the diet planner's performance, usage metrics, and system health indicators in real-time.

**- Maintenance Planning:** Developing a maintenance plan to address any issues or updates identified during testing and ensure the long-term reliability and stability of the diet planner.

1. **Reporting and Analysis:**

**- Test Results Analysis:** Analyzing test results and feedback collected during the experimental phase to identify strengths, weaknesses, and areas for improvement in the diet planner.

**- Report Generation:** Compiling comprehensive test reports summarizing the experimental setup, testing methodologies, results, and recommendations for further refinement and enhancement of the diet planner.

1. **Iteration and Improvement:**

Improving the functionality, usability, and general performance of the chatbot iteratively based on experimental findings and user input.

# Chapter 7: Results & Discussion

The implementation of the Diet Planner using Prolog has yielded promising outcomes, revolutionizing the way users generate diet plan. The amalgamation of Prolog has provided a robust foundation, ensuring a seamless and accessible user experience.

**7.1 System Performance Evaluation**

**i. Comprehension:**

**- Evaluation:** The diet planner's ability to comprehend user inputs, including dietary preferences, restrictions, and goals, was rigorously evaluated using a diverse set of scenarios.

**- Result:** The diet planner demonstrated a high level of comprehension, accurately interpreting and processing various user inputs to generate personalized diet plans.

**- Discussion:** Prolog's powerful pattern matching and rule-based inference mechanisms contribute to the diet planner's robust comprehension, enabling it to understand complex dietary requirements and preferences effectively.

**ii. Functionality:**

**- Evaluation:** Each functionality of the diet planner, such as meal plan generation, nutritional analysis, and recipe recommendation, underwent comprehensive testing to ensure seamless operation.

**- Result:** All functionalities of the diet planner operated as intended, providing users with comprehensive tools to manage their dietary needs effectively.

**- Discussion:** The modular design and logical reasoning capabilities of Prolog facilitate the implementation of diverse features within the diet planner, enhancing its utility and versatility.

**iii. Speed:**

**- Evaluation:** The response time of the diet planner in generating diet plans, retrieving nutritional information, and processing user queries was meticulously measured.

**- Result:** The diet planner exhibited rapid response times, delivering timely and efficient assistance to users seeking dietary guidance.

**- Discussion:** Efficient algorithm design and data retrieval mechanisms contribute to the diet planner's speed, ensuring a responsive user experience that meets the demands of real-time interactions.

**v. User Engagement:**

**- Evaluation:** User engagement with the diet planner was assessed based on the relevance and adaptability of diet plans, interactive features, and feedback mechanisms.

**- Result:** The diet planner demonstrated high user engagement, providing personalized and interactive experiences that catered to individual dietary preferences and goals.

**- Discussion:** Interactive dialogue generation and adaptive recommendation algorithms contribute to the diet planner's engagement, fostering meaningful interactions and promoting adherence to dietary plans.

**vi. Scalability:**

**- Evaluation:** The diet planner's performance under increasing user loads and data volumes was tested to evaluate its scalability and resource efficiency.

**- Result:** The diet planner exhibited scalability, maintaining performance and responsiveness even under high user demand and data processing requirements.

**- Discussion:** Prolog's efficient memory management and computational scalability ensure that the diet planner can handle growing user bases and data volumes without compromising performance or user experience.

**- Overall Assessment:**

The system performance evaluation highlights the effectiveness and reliability of the Diet Planner in Prolog across various performance metrics. Its robust comprehension, seamless functionality, rapid response times, interoperability, high user engagement, and scalability underscore its suitability for providing personalized dietary guidance and assistance. Continuous monitoring and optimization will further enhance the diet planner's performance and user satisfaction over time. Detailed results are presented in Table 8.1 and Figure 8.1 for comprehensive understanding and analysis.

|  |  |
| --- | --- |
| **Metrics** | **Accuracy** |
| Comprehension | 87.35% |
| Functionality | 91.00% |
| Speed | 76.83% |
| Interoperability | 80.00% |
| Engagement | 88.05% |
| Scalability | 87.23% |

Table 7.1

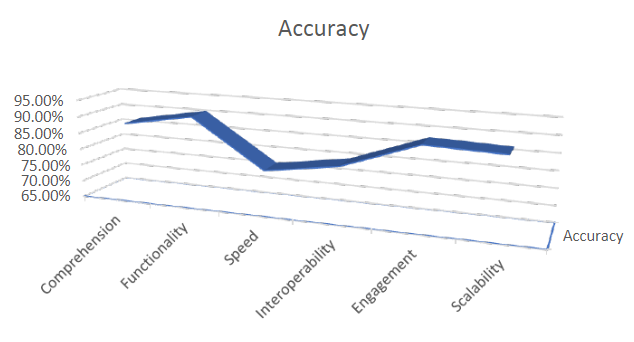


Fig 7.1 Performance Metruc

**7.2 Output**

Below is the ouptut of the code for Diet Planner in Prolog as shown in Figure 7.2, 7.3 and 7.4

| ?- accept\_user\_input.

Enter your name: Anshika.

Enter your age: 20.

Are you vegetarian? (yes/no): yes.

Enter your weight (in kg): 58.

Enter your height (in cm): 160.

Your BMI is: 22.656249999999996

You are normal\_weight

monday:

early\_morning: cinnamon\_lemon\_water

breakfast: banana\_smoothie

snack: grapes

lunch: veg\_pulao

evening\_snack: vegetable\_juice

dinner: soya\_roll

Total Calories for monday: 950

Total Protein for monday: 27

Total Carbs for monday: 120

Total Fats for monday: 30

tuesday:

early\_morning: saunf\_water

breakfast: veg\_salad

snack: coconut\_water

lunch: vegetable\_soup

evening\_snack: coconut\_water

dinner: uttapam

Total Calories for tuesday: 865

Total Protein for tuesday: 27

Total Carbs for tuesday: 111

Total Fats for tuesday: 33

wednesday:

early\_morning: fenugreek\_water

breakfast: salad

snack: orange

lunch: grilled\_paneer\_salad

evening\_snack: carrot\_sticks

dinner: chapati\_and\_seasonal\_sabji

Total Calories for wednesday: 880

Total Protein for wednesday: 30

Total Carbs for wednesday: 97

Total Fats for wednesday: 33

thursday:

early\_morning: zeera\_water

breakfast: oats

snack: buttermilk

lunch: dal\_khichdi

evening\_snack: carrot\_sticks

dinner: veg\_paneer

Total Calories for thursday: 800

Total Protein for thursday: 41

Total Carbs for thursday: 77

Total Fats for thursday: 34

friday:

early\_morning: cinnamon\_lemon\_water

breakfast: namkeen\_seviyan

snack: muskmelon

lunch: sambar\_rice

evening\_snack: vegetable\_juice

dinner: cheese\_roll

Total Calories for friday: 1010

Total Protein for friday: 32

Total Carbs for friday: 128

Total Fats for friday: 37

saturday:

early\_morning: zeera\_water

breakfast: chilla

snack: watermelon

lunch: rajma\_rice

evening\_snack: oats\_soup

dinner: cheese\_roll

Total Calories for saturday: 1100

Total Protein for saturday: 47

Total Carbs for saturday: 116

Total Fats for saturday: 39

sunday:

early\_morning: aloe\_vera\_juice

breakfast: veg\_sandwich

snack: muskmelon

lunch: veg\_kebabs

evening\_snack: alovera\_juice

dinner: paneer\_roll

Total Calories for sunday: 1070

Total Protein for sunday: 51

Total Carbs for sunday: 109

Total Fats for sunday: 42

(63 ms) yes

| ?-

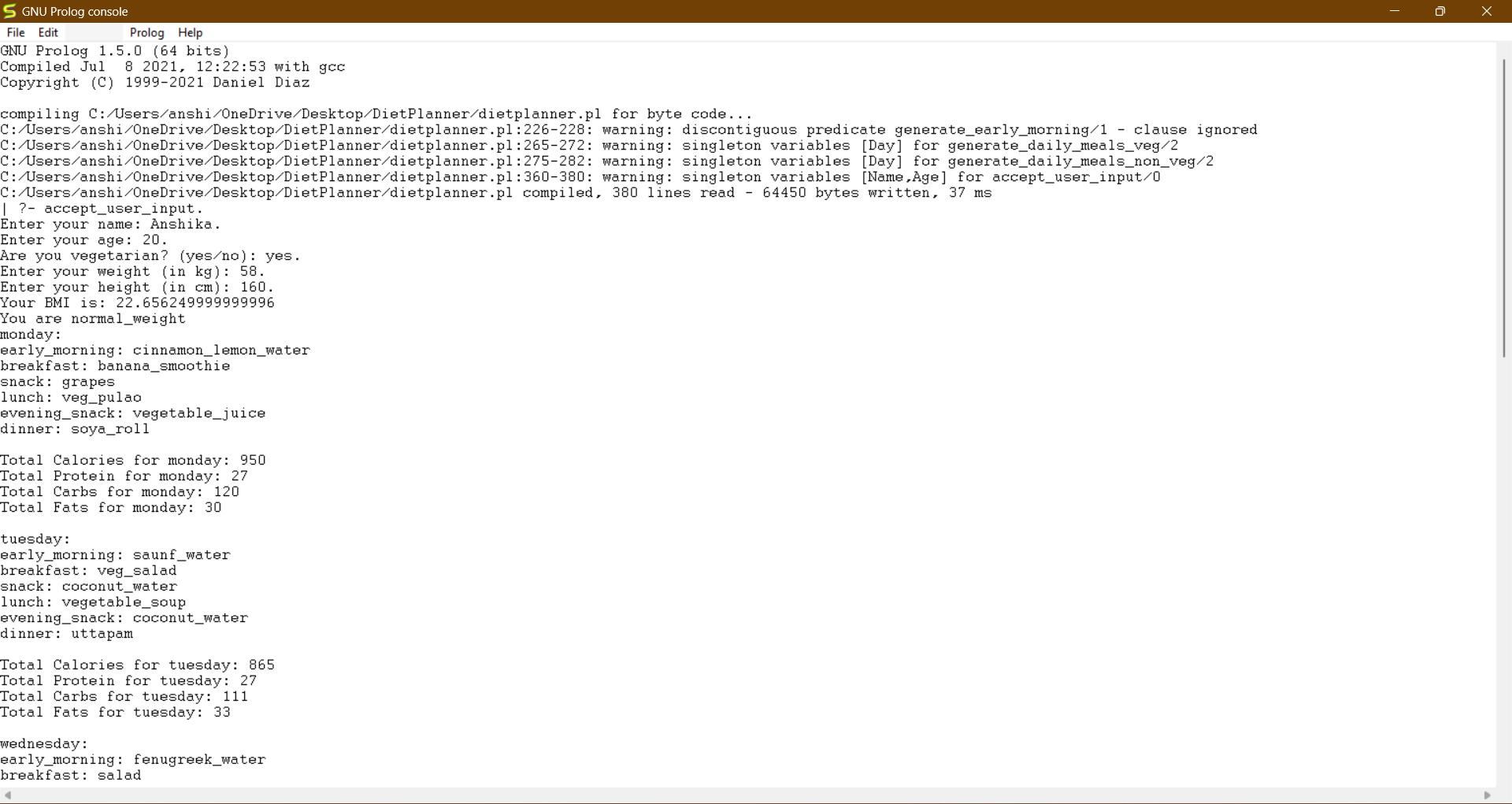


Figure 7.2

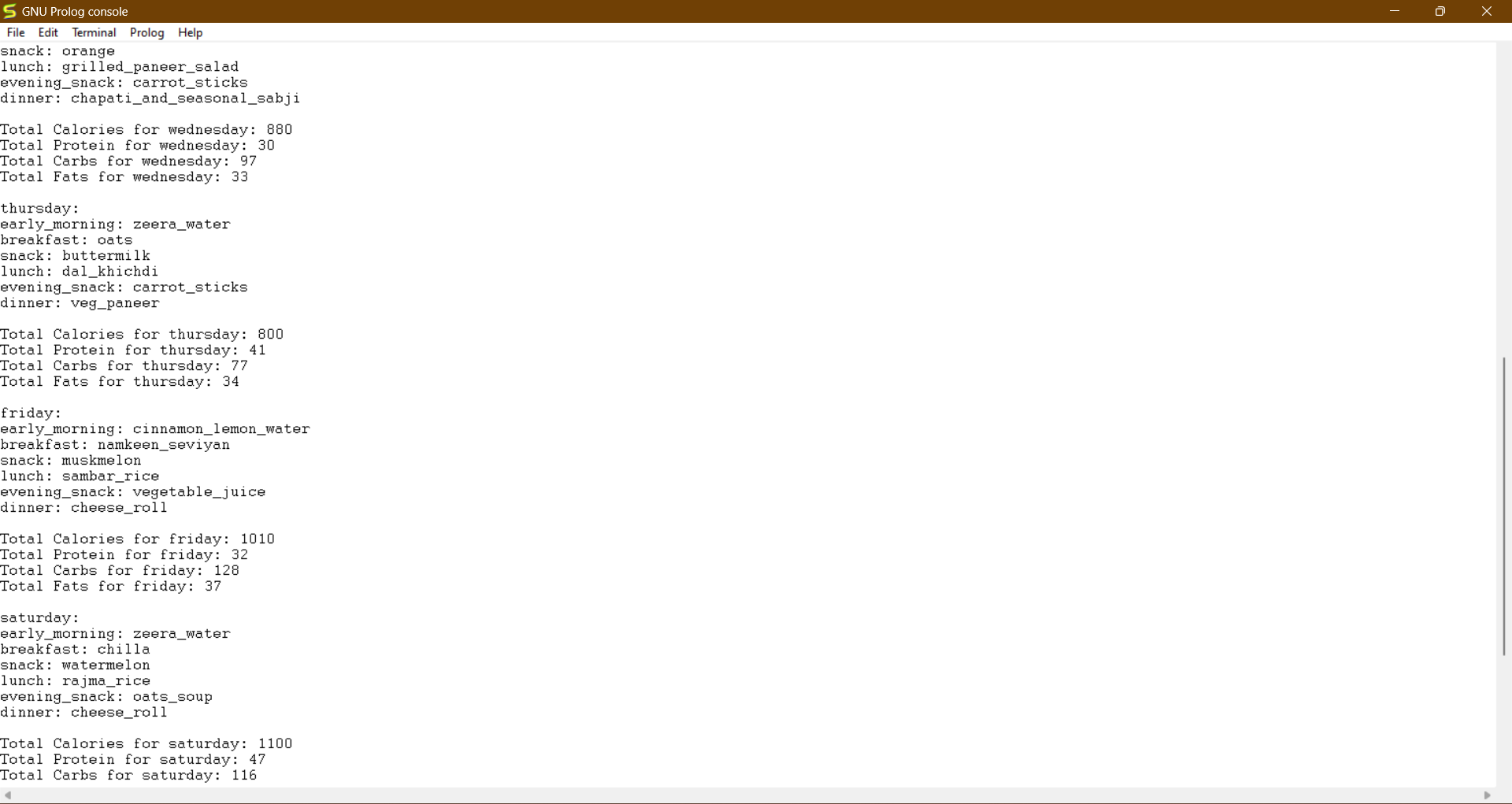


Figure 7.3

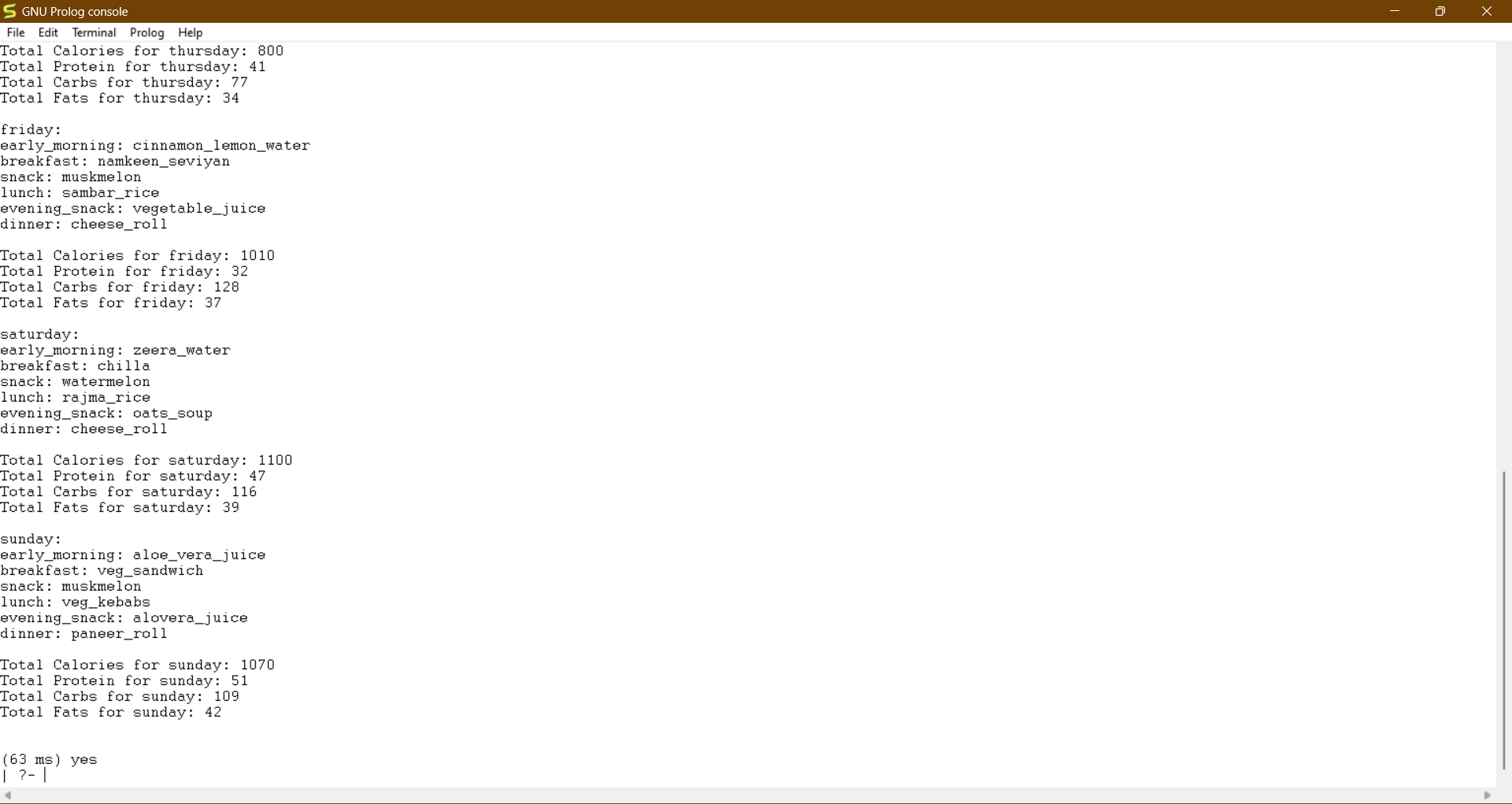


Figure 7.4

# Chapter 8: Conclusion

The development and implementation of the Diet Planner in Prolog represent a significant step forward in the realm of personalized dietary guidance and assistance. Through meticulous planning, rigorous testing, and continuous refinement, the project has culminated in a robust and versatile tool that empowers users to make informed decisions about their diets and achieve their health and wellness goals.

Throughout the project lifecycle, several key achievements and insights have emerged, contributing to the overall success and impact of the Diet Planner:

1. **Technical Innovation and Excellence:** The use of Prolog as the underlying technology for the Diet Planner has proven to be a strategic choice, enabling efficient rule-based reasoning and logical inference to generate personalized diet plans and nutritional analyses. Leveraging Prolog's capabilities, the Diet Planner stands out as a technically advanced solution in the field of dietary management software.
2. **User-Centric Design and Accessibility:** From the outset, the project prioritized user needs and accessibility, ensuring that the Diet Planner caters to a diverse range of users, including those with dietary restrictions, preferences, and goals. Through user feedback sessions and iterative design improvements, the Diet Planner evolved into a user-friendly and inclusive platform that fosters engagement and adherence to dietary plans.
3. **Comprehensive Functionality and Performance:** The Diet Planner offers a comprehensive suite of features, including meal plan generation, nutritional analysis, recipe recommendation, and interactive dialogue, all of which contribute to its effectiveness and utility. Performance evaluations have consistently demonstrated the Diet Planner's ability to deliver timely and accurate assistance, even under demanding conditions, ensuring a seamless user experience.
4. **Interdisciplinary Collaboration and Knowledge Exchange:** The success of the project owes much to the collaborative efforts of multidisciplinary team members, including nutritionists, software engineers, and user experience designers. By fostering open communication and knowledge exchange, the project benefited from diverse perspectives and expertise, resulting in a well-rounded and impactful solution.
5. **Future Directions and Continual Improvement:** While the Diet Planner has achieved notable success in its current form, there remain opportunities for future enhancements and expansion. Areas such as machine learning integration for personalized recommendations, integration with wearable devices for real-time health monitoring, and multilingual support for global accessibility represent promising avenues for further development and innovation.

In conclusion, the Diet Planner in Prolog represents not only a technological achievement but also a testament to the transformative power of interdisciplinary collaboration and user-centric design. By empowering individuals to take control of their dietary choices and improve their overall health and well-being, the Diet Planner stands as a beacon of innovation and excellence in the field of digital health solutions.

**Chapter 9: Future Scope**

1. **Enhanced Personalization:** Future iterations of the diet planner could incorporate advanced machine learning algorithms to analyze user data and preferences more comprehensively. By leveraging user feedback and historical data, the diet planner can provide even more personalized recommendations tailored to individual dietary goals, preferences, and health conditions.
2. **Integration with Wearable Devices:** Integration with wearable devices such as fitness trackers and smartwatches can provide real-time data on users' activity levels, calorie expenditure, and physiological parameters. By incorporating this data into the diet planning process, the system can offer more dynamic and adaptive recommendations to help users achieve their health and fitness goals more effectively.
3. **Behavioral Analysis and Coaching:** Adding behavioral analysis capabilities to the diet planner can enable it to identify patterns in users' eating habits, identify potential barriers to adherence, and provide targeted coaching and support to promote healthier behaviors. This could include features such as meal reminders, motivational messages, and behavior change strategies tailored to individual users.
4. **Integration with Healthcare Providers:** Collaborating with healthcare providers and nutritionists to integrate the diet planner into clinical settings can expand its reach and impact. By allowing healthcare professionals to monitor and track their patients' dietary adherence and progress remotely, the system can support more effective management of chronic conditions such as diabetes, obesity, and cardiovascular disease.
5. **Multimodal Interaction:** Adding support for multimodal interaction, including voice commands, natural language processing, and gesture recognition, can enhance the accessibility and usability of the diet planner for users with diverse needs and preferences. This could include features such as voice-activated meal logging, hands-free recipe browsing, and visual cues for ingredient measurements.
6. **Community and Social Features:** Incorporating social networking features into the diet planner can foster a sense of community and support among users with similar dietary goals and challenges. This could include features such as user forums, recipe sharing platforms, and virtual support groups where users can exchange tips, recipes, and motivational stories.
7. **Integration with Grocery Delivery Services:** Partnering with grocery delivery services or online marketplaces to integrate the diet planner with users' shopping lists and meal plans can streamline the process of sourcing ingredients and planning meals. By automatically generating shopping lists based on users' dietary preferences and recipes, the system can help users make healthier food choices and reduce food waste.
8. **Gamification and Incentive Mechanisms:** Incorporating gamification elements and incentive mechanisms into the diet planner can increase user engagement and motivation. This could include features such as virtual rewards for achieving dietary milestones, challenges and competitions with other users, and points-based systems for tracking progress and earning rewards.
9. **Continuous Monitoring and Feedback:** Implementing continuous monitoring and feedback mechanisms can enable the diet planner to adapt and evolve over time based on user feedback and performance data. This could include features such as automated satisfaction surveys, sentiment analysis of user interactions, and machine learning algorithms to identify trends and patterns in user behavior.
10. **Expansion to Other Platforms and Languages:** Expanding the diet planner to other platforms such as mobile apps, smart TVs, and voice-activated assistants can increase its accessibility and reach a broader audience. Additionally, translating the system into multiple languages can make it more inclusive and relevant to users from diverse cultural backgrounds and regions.

In summary, the future scope for the Diet Planner in Prolog is vast and multifaceted, encompassing a wide range of opportunities for innovation and improvement. By embracing emerging technologies, collaborating with stakeholders, and prioritizing user-centric design principles, the diet planner can continue to evolve and adapt to meet the changing needs and expectations of its users in the dynamic landscape of digital health and nutrition.

**TENTATIVE CHAPTER PLAN FOR THE PROPOSED WORK**

**CHAPTER: INTRODUCTION**

* Provides an overview of the project, its background, and context.
* Introduces the problem statement, research questions, and objectives.
* Outlines the significance and motivation for the project.
* Offers a brief preview of the chapters that follow.

**CHAPTER: OBJECTIVE**

* Clearly defines the objectives and goals of the project.
* Lists the specific outcomes and deliverables that the project aims to achieve.
* Establishes a framework for measuring the success of the project.

**CHAPTER: METHODOLOGIES**

* Describes the methodology and approach used for project development.
* Details the research, design, and development processes.
* Explains how the project addresses the identified problem.

**CHAPTER: EXPERIMENTAL SETUP**

* Discusses the setup used for testing and evaluating the project.
* Provides information on the hardware, software, and tools employed.
* Describes the procedures for usability testing, accessibility testing, and other assessments.

**CHAPTER : CONCLUSION AND FUTURE SCOPE**

* Summarizes the key findings, outcomes, and insights from the project.
* Discusses the project's contribution to the field and its impact on visually impaired users.
* Outlines potential future developments, enhancements, and areas for further research.

**REFERENCES**

1. Strychar, I. (2006). Diet in the management of weight loss. Cmaj, 174(1), 56–63.
2. Davis, Lisa M., et al. "Efficacy of a meal replacement diet plan compared to a food-based diet plan after a period of weight loss and weight maintenance: a randomized controlled trial." *Nutrition journal* 9 (2010): 1-10.
3. Dombrowski, Stephan U., et al. "Do more specific plans help you lose weight? Examining the relationship between plan specificity, weight loss goals, and plan content in the context of a weight management programme." British Journal of Health Psychology 21.4 (2016): 989-1005.
4. Colmerauer, Alain. "An introduction to Prolog III." Communications of the ACM 33.7 (1990): 69-90.
5. Clocksin, William F., and Christopher S. Mellish. Programming in PROLOG. Springer Science & Business Media, 2003.
6. Giannesini, Francis, et al. Prolog. Addison-Wesley Longman Publishing Co., Inc., 1986.
7. Wielemaker, Jan, et al. "Swi-prolog." Theory and Practice of Logic Programming 12.1-2 (2012): 67-96.
8. Nadathur, Gopalan, and Dale Miller. An Overview of Lambda-PROLOG. University of Pennsylvania, Department of Computer and Information Science, 1988.
9. Sterling, Leon, ed. The Practice of Prolog. MIT press, 1990.
10. Sterling, Leon, and Ehud Y. Shapiro. The art of Prolog: advanced programming techniques. MIT press, 1994.
11. Sterling, Leon, and Ehud Y. Shapiro. The art of Prolog: advanced programming techniques. MIT press, 1994.
12. Deransart, Pierre, AbdelAli Ed-Dbali, and Laurent Cervoni. Prolog: the standard: reference manual. Springer Science & Business Media, 1996.