AHP: The Analytic Hierarchy Process

A project report submitted in partial fulfilment of the requirement for the Award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

By

AIRLA SNEHITH (160720733091)

MOHAMMED KHUBAIB (160720733092)

ASHISH KUMAR PANDIA (160720733093)

Under the Guidance of

Mr. Shaik Kareem Basha, Assistant Professor, Dept. of CSE

Department of Computer Science and Engineering

Methodist College of Engineering and Technology

King Koti, Abids, Hyderabad-500001.

2022-2023

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001,

Department of Computer Science and Engineering

DECLARATION BY THE CANDIDATES

We, **AIRLA SNEHITH(160720733091), MOHAMMED KHUBAIB(160720733092) and ASHISH KUMAR PANDIA(160720733093**) students of Methodist College of Engineering and Technology, pursuing Bachelor's degree in Computer Science and Engineering, here by declare that this project report entitled "**AHP:The Analytic Hierarchy Process",** carried out under the guidance of **Mr. Shaik Kareem Basha** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer Science. This is a record work carried out by us and the results embodied in this project have not been reproduced/copied from any source.

AIRLA SNEHITH(160720733091)

MOHAMMED KHUBAIB(160720733092)

ASHISH KUMAR PANDIA(160720733093)

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.

Department of Computer Science and Engineering

CERTIFICATE BY THE SUPERVISOR

This is to certify that this project report entitled "**AHP:The Analytic Hierarchy Process**" being submitted *by* **Airla Snehith**(160720733091), **Mohammed Khubaib**(160720733092) and **Ashish Kumar Pandia**(160720733093), submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering, during the academic year 2022-23, is a bonafide record of work carried out by them.

Mr. Shaik Kareem Basha

Assistant Professor,

Date:

Methodist College of Engineering and Technology,
King Koti , Abids , Hyderabad-500001.

Department of Computer Science and Engineering

CERTIFICATE BY THE HEAD OF THE DEPARTMENT

This is to certify that this project report entitled "**AHP:The Analytic Hierarchy Process**" *by* **Airla Snehith**(160720733091), **Mohammed Khubaib**(160720733092) and **Ashish Kumar Pandia**(160720733093),submitted in partial fulfillment of the requirements for the degreeof Bachelor of Engineering in Computer Science and Engineering of the Osmania University,Hyderabad, during the academic year 2022-23, is a bonafide record of work carried out by them.

Mrs. P. Lavanya
Professor &
Head of the Department.

Date:

ACKNOWLEDGEMNTS

We would like to express our sincere gratitude to my project guide **Mrs. P. Lavanya, Assistant Professor,** for giving us the opportunity to work on this topic. It would never be possible for us to take this project to this level without his innovative ideas and his relentless support and encouragement.

We would like to thank our project coordinator **Mr. Shaik Kareem Basha, Assistant Professor,** who helped us by being an example of high vision and pushing towards greater limits of achievement.

Our sincere thanks to **Mrs. P. Lavanya, Professor** and **Head of the Department of Computer Science and Engineering,** for her valuable guidance and encouragement which has played a major role in the completion of the project and for helping us by being an example of high vision and pushing towards greater limits of achievement.

We would like to express a deep sense of gratitude towards the **Dr. G. Ravinder Reddy, Principal, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We would like to express a deep sense of gratitude towards the **Dr. Lakshmipathi Rao, Director, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We are indebted to the Department of Computer Science & Engineering and Methodist College of Engineering and Technology for providing us with all the required facility to carry our work in a congenial environment. We extend our gratitude to the CSE Department staff for providing us to the needful time to time whenever requested.

We would like to thank our parents for allowing us to realize our potential, all the support they have provided us over the years was the greatest gift anyone has ever given us and also for teaching us the value of hard work and education. Our parents has offered us with tremendous support and encouragement, thanks to our parents for all the moral support and the amazing opportunities they have given us over the years.

ABSTRACT

The Analytic Hierarchy Process (AHP)

The Analytic Hierarchy Process (AHP) is a powerful and widely used **Decision-making methodology** developed by **Dr. Thomas L. Saaty** in the **1970s**. It provides a structured and systematic approach to tackle complex decisions by breaking them down into a hierarchical structure of criteria and options, enabling individuals and organizations to make informed and rational choices.

The AHP methodology begins by defining the ultimate goal or objective of the decision-making process. Next, a hierarchical structure is constructed, consisting of criteria that are essential in achieving the objective and the available options that must be evaluated. The criteria and options are organized in a top-down fashion, with the primary goal at the top and sub-criteria and sub-options branching below.

Once the hierarchy is established, the decision-makers proceed to a pair-wise comparison phase. They systematically compare the relative importance of each criterion and option against others, assigning numerical values to represent the preference relationships.

The pair-wise comparison data is then synthesized using the Eigenvector method to calculate the relative weights of the criteria and options. These weights represent the importance of each element in contributing to the overall objective, thus providing a quantitative basis for decision-making.

The strength of the AHP lies in its ability to handle subjective, qualitative, and quantitative factors concurrently. It accommodates individual preferences and integrates expert judgments while maintaining a structured and transparent decision-making process.

In **conclusion**, the Analytic Hierarchy Process (AHP) offers a systematic and robust approach to make well-informed decisions in complex and multi-criteria situations. By incorporating subjective judgments, quantitative data, and mathematical principles, the AHP has become a valuable tool in a wide range of decision-making scenarios. Its ability to simplify complex problems and provide coherent and rational choices has made it an indispensable methodology for individuals and organizations striving for optimal outcomes.

Table of Contents

List of Figures

1. Introduction to the Analytic Hierarchy Process (AHP)

In today's complex and fast-paced world, decision-making is a ubiquitous challenge that individuals and organizations face on a regular basis. The ability to make sound and well-informed decisions is crucial for success, be it in business, project management, personal life, or any other domain. However, with the abundance of information, numerous criteria to consider, and a wide array of available options, decision-making can become overwhelming and prone to biases.

The Analytic Hierarchy Process (AHP) Web Application comes as a transformative solution to streamline decision-making processes and enhance the quality of choices. Developed as a user-friendly and powerful tool, the AHP Web Application harnesses the principles of the Analytic Hierarchy Process, a widely respected decision-making technique introduced by **Dr. Thomas L. Saaty in the 1970s.**

AHP is a structured and systematic approach that breaks down complex decisions into a hierarchical structure of criteria and options, allowing decision-makers to evaluate and prioritize choices based on their relative importance. This methodology not only simplifies the decision-making process but also promotes consistency, rationality, and transparency in the evaluation of alternatives.

The objective of the AHP Web Application is to empower users with a straightforward yet robust tool for making informed decisions in diverse scenarios. Whether it's a business executive evaluating potential investment opportunities, a project manager choosing the best course of action, a student determining which college to attend, or an individual making a personal life decision, the AHP Web Application can provide valuable insights and guidance.

To ensure a seamless and efficient user experience, the AHP Web Application is built using modern web development technologies. Leveraging the capabilities of Streamlit, a Python library known for its simplicity and effectiveness in building web applications, users can easily access and navigate the application without any technical hurdles. Furthermore, the AHP algorithm, a fundamental component of the application, is implemented using the powerful numerical computation libraries, NumPy and Pandas, to ensure accurate and reliable results.

The AHP Web Application operates on a step-by-step approach, guiding users through the decision-making process. Users begin by defining their ultimate goal, followed by listing the relevant criteria and available options. Through a pair-wise prioritization process, users establish the relative importance of each criterion and option, enabling the AHP algorithm to compute the optimal choice based on the user's input.

In conclusion, the Analytic Hierarchy Process (AHP) Web Application serves as an indispensable tool in contemporary decision-making, empowering users to tackle complex choices with confidence and clarity. Its user-friendly interface, combined with the power of the AHP methodology, makes it a versatile tool applicable in various contexts, ranging from individual decisions to organizational strategies. With the AHP Web Application at their disposal, users can unlock the potential to make more informed and rational decisions, thus shaping a path towards success and prosperity.

2. LITERATURE SURVEY

Literature Survey for Analytic Hierarchy Process (AHP) Web Application:

2.1 The Analytic Hierarchy Process: A Comprehensive Literature Review

This literature review provides an extensive survey of the Analytic Hierarchy Process (AHP) across various domains, including business, engineering, environmental management, and decision-making. It explores the development of AHP, its applications, and case studies. The review highlights the effectiveness and flexibility of AHP as a decision-making tool and emphasizes its utility in handling complex problems with multiple criteria and alternatives.

2.2 Web-Based Decision Support System Using AHP for Project Portfolio Management

This research introduces a web-based decision support system that leverages AHP for project portfolio management. The system allows stakeholders to define goals, criteria, and options, and perform pair-wise comparisons to prioritize projects based on their alignment with organizational objectives. The study demonstrates how web applications can enhance AHP's accessibility and usability in project

2.3 AHP-Based E-Learning System for Course Selection

This study presents an AHP-based e-learning system to assist students in making informed decisions for course selection. The web application enables students to define their educational goals, evaluate course criteria, and compare available courses. The AHP algorithm then recommends the most suitable courses based on students' preferences and priorities.

2.4 Web-Based Decision Support System for Vendor Selection Using AHP

This research proposes a web-based decision support system for vendor selection in supply chain management using AHP. The system facilitates the comparison of vendors based on multiple criteria, such as cost, quality, and delivery time. Users can input their preferences, and the AHP algorithm generates recommendations for the most suitable vendor.

2.5 AHP-Web: A Web-Based Multi-Criteria Decision Analysis Tool

AHP-Web is a user-friendly web application that simplifies the application of AHP in decision-making processes. The tool allows users to define decision hierarchies, set criteria weights through pair-wise comparisons, and analyze the results. It provides an intuitive interface to streamline the AHP process and make it accessible to a broader audience.

2.6 Web-Based AHP Approach for Supplier Evaluation in Green Supply Chain Management

This study presents a web-based AHP approach for supplier evaluation in green supply chain management. The system assists organizations in evaluating and selecting suppliers based on environmental and sustainability criteria. The web application offers a straightforward method for incorporating sustainability considerations into the decision-making process.

These research works demonstrate the diverse applications of the Analytic Hierarchy Process in web-based decision support systems and the benefits of making AHP accessible through web applications. The AHP Web Application developed in this project contributes to this growing body of research by providing a simple yet powerful tool for decision-making using AHP in various domains.

3. DESIGN ANALYSIS

3.1 Data Flow Diagram:

- A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects.

- A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated.

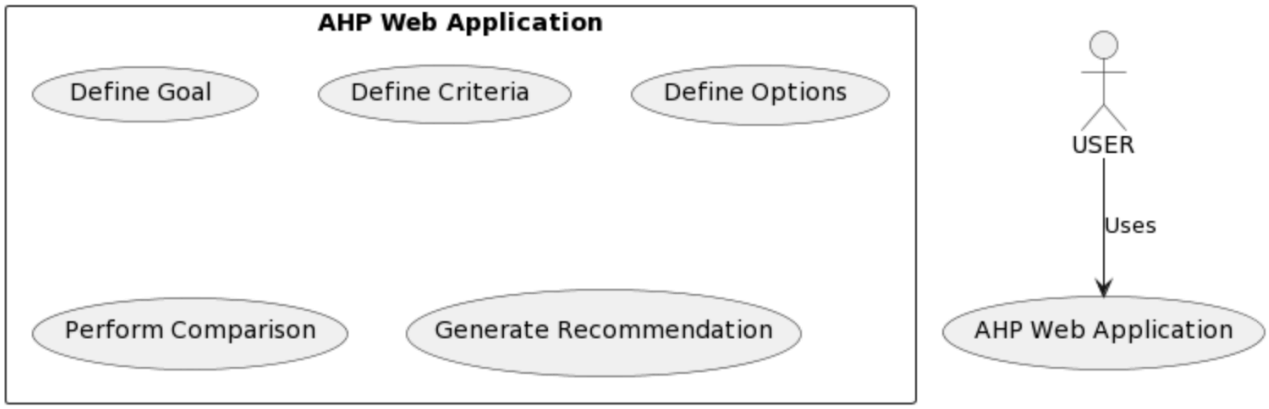


Figure 3.1: DFD

3.2 UML Diagrams

UML stands for unified modeling language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standardized is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. UML is comprised of two major components: a Mta-model and a notation. In the future, some form of method or process may also be added to; or association with UML.It is important to distinguish between the UML model and the set of diagrams of a system. A diagram is a partial graphic representation of a system's model. The set of diagrams need not completely cover the model and deleting a diagram does not change the model. The model may also contain documentation that drives the model elements and diagrams. In 1997 UML was adopted as a standard by the Object Management Group (OMG), and has been managed by this organization ever since. In 2005 UML was also published by the International Organization for Standardization (ISO) as an approved ISO standard. Since then it has been periodically revised to cover the latest revision of UML.

UML is not a development method by itself; however, it was designed to be compatible with the leading object-oriented software development methods of its time, for example OMT, Booch method, Objectory and especially RUP that it was originally intended to be used with when work began at Rational Software.UML (Unified Modeling Language) is a standard notation for the modeling of real-world objects as a first step in developing an object-oriented design methodology. The major perspectives of a UML are Design, Implementation, Process and Deployment. The center is the Use Case view which connects all these four. A Use case represents the functionality of the system.

3.2.1 Use Case diagram

- Use Case diagrams in the UML is a type of behaviour diagrams defined by and created from a Use-case analysis.
- It is used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).
- The main purpose of a Use case diagram is to show whar system functions are performed for which actor.
- Use case diagrams are drawn to capture the functional requirements of a system.

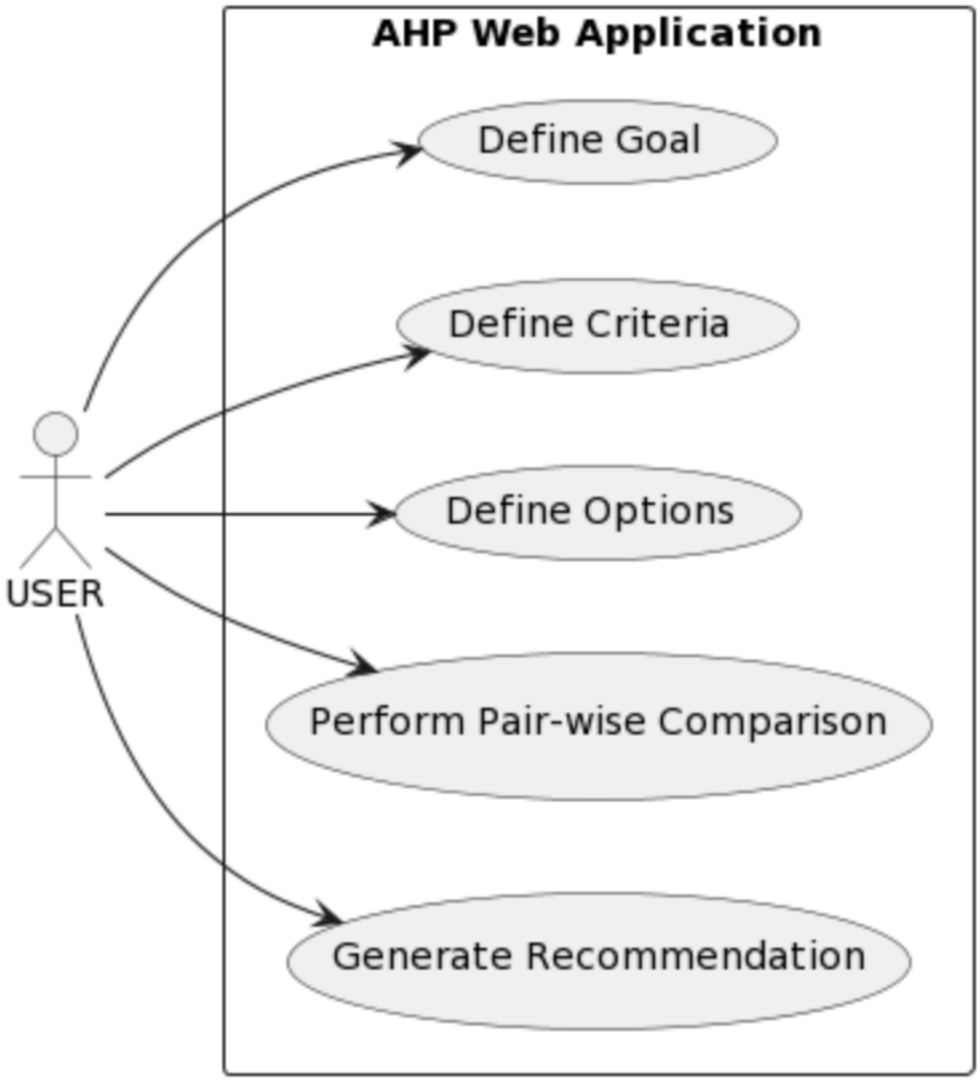


Figure 3.2.1: Use Case Diagram

3.2.2 Class Diagram

- Class Diagram is a type of static structure diagram that describes the structure of a system.
- It shows the system's classes, their attributes, operations (or methods), and the relationships among objects.
- It explains which class contains information.

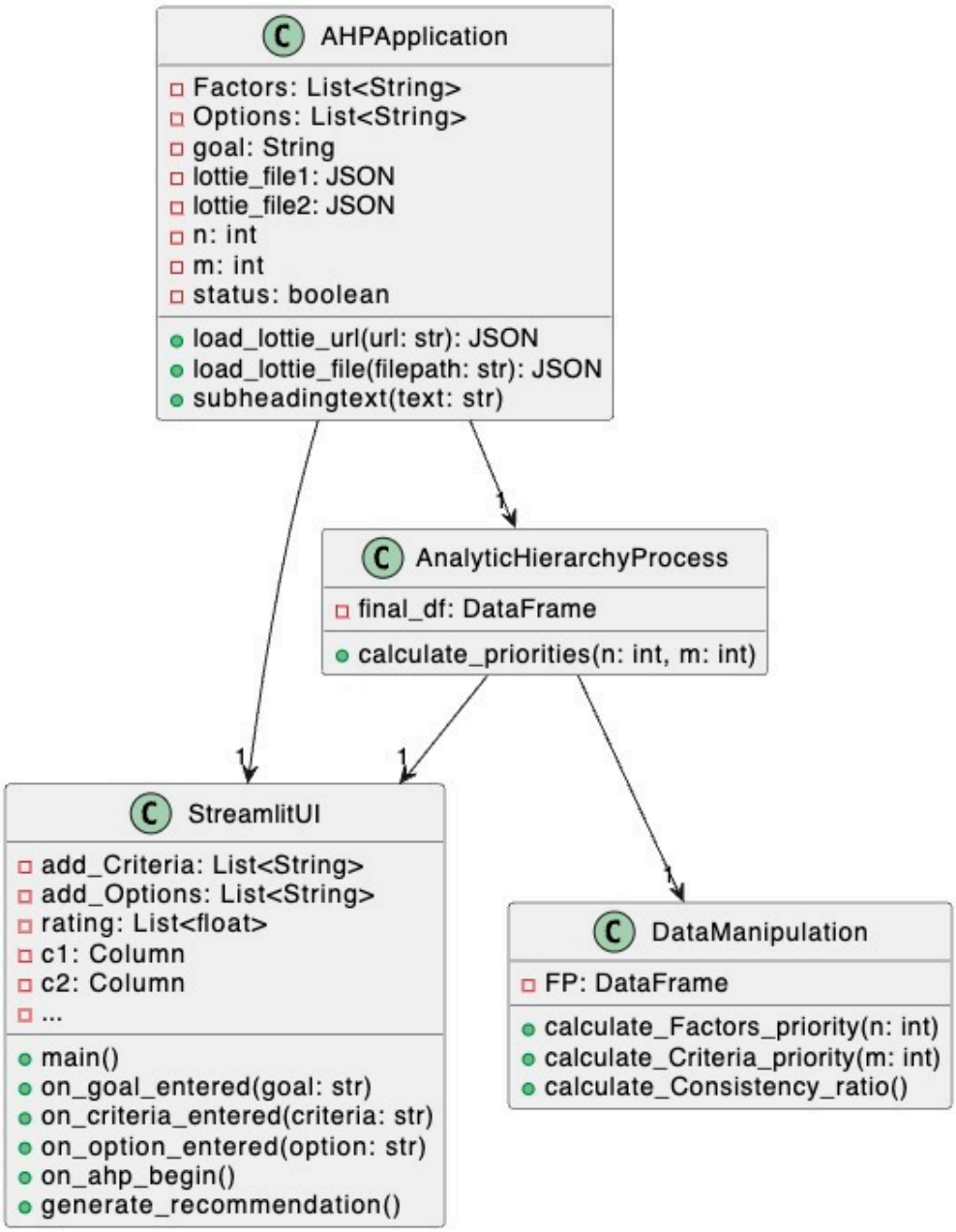


Figure3.2.2: Class Diagram

3.2.3 Sequence Diagram:

- A sequence diagram shows object interactions arranged in time sequence.
- It depicts objects and classes involved in the scenario and the sequence of messages exchanged between the objectsneeded to carry out the functionality of the scenario.
- Sometimes called as event diagrams,event scenarios, timing diagram.
- A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order.

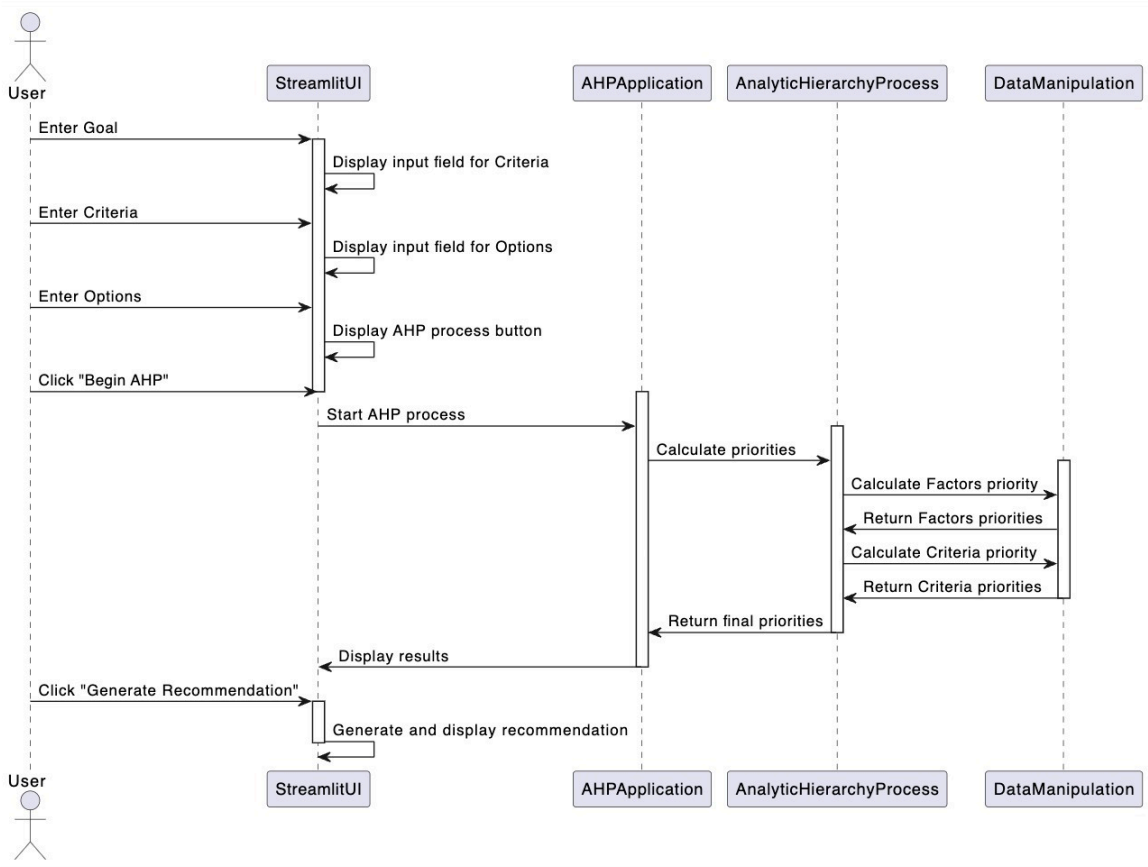


Figure 3.2.3: Sequence Diagram

3.2.4 Activity Diagram:

Activity diagram is basically a flow chart to represent the flow from one activity to another activity.The activity can be described as an operation of the system. So the control flow is drawn from one operation to another.It can be used to describe the business and operational step-by-step workflows of components in the system.

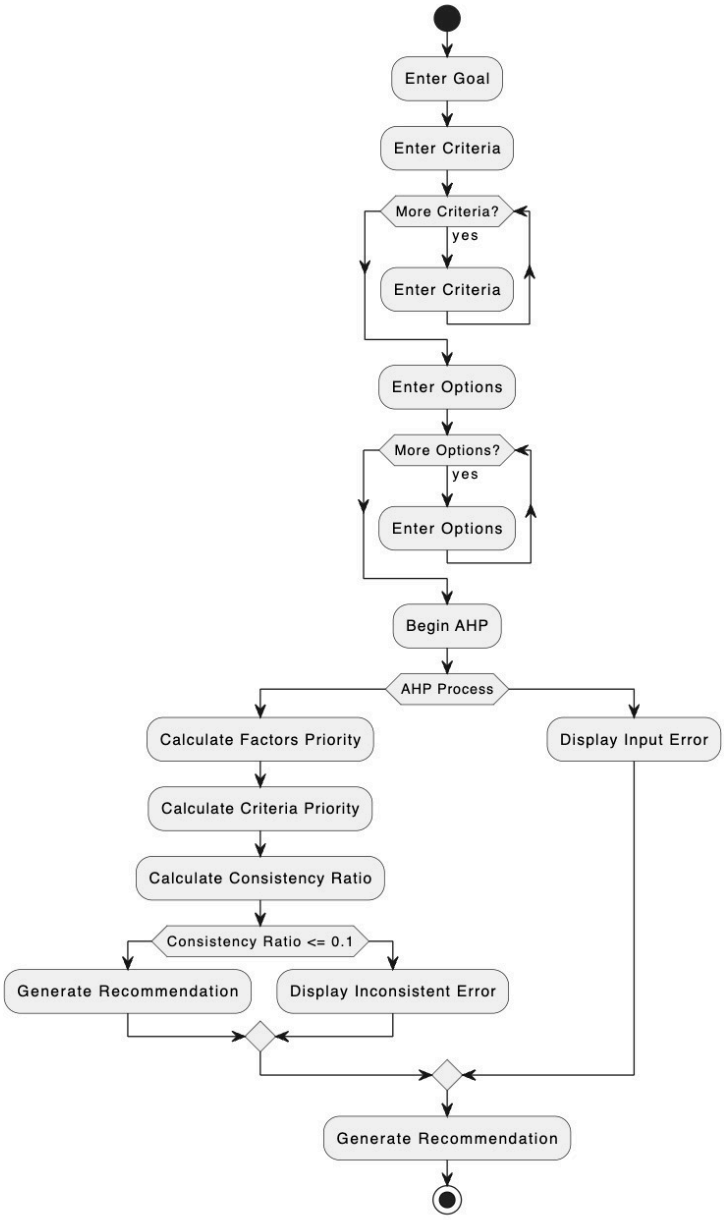


Figure 3.2.4: Activity diagram

3.2.5 Component Diagram:

A Component Diagram is a type of UML diagram that provides a visual representation of the high-level components or building blocks of a software system and their interactions. It emphasizes the modular structure and encapsulation of the system, showing how components are connected through interfaces. Each component represents a distinct, self-contained unit of functionality, and they can be implemented independently and then integrated to create the complete system.

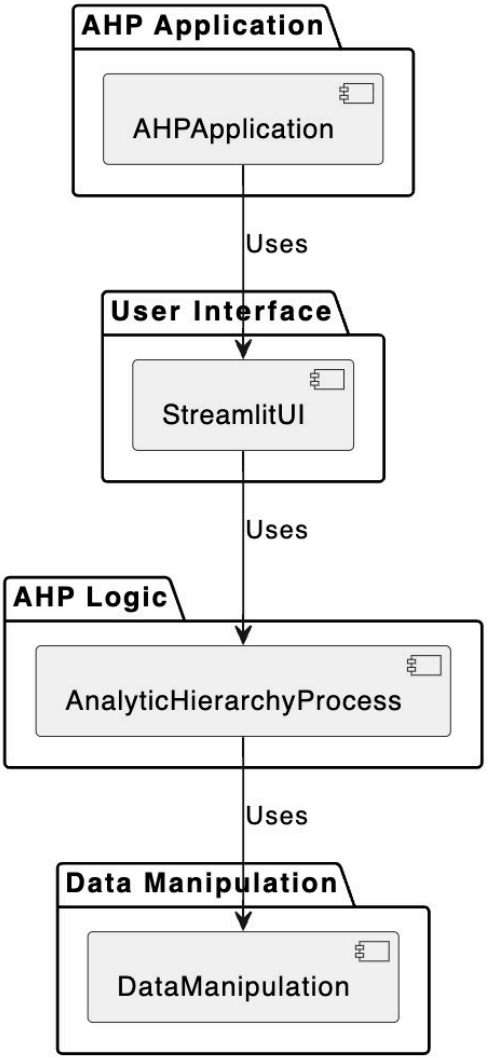


Figure 3.2.5: Component diagram

4. Implementation

4.1 MODULES : AHP Implementation

Module Description:

The "AHP Implementation" module is the core component responsible for performing the Analytic Hierarchy Process (AHP) algorithm to prioritize criteria and options based on user inputs. It takes the data provided by the user through the AHP Web Application's user interface and processes it to generate a recommendation for the best option.

4.2 Detailed Implementation:

4.2.1 Input Data Collection:

The module receives the user inputs from the AHP Web Application, which include the defined goal, criteria, and options. These inputs are collected in data structures for further processing.

4.2.2 Pair-wise Comparison:

The AHP algorithm requires pair-wise comparisons of criteria and options to determine their relative importance. The module presents the user with pair-wise comparison matrices for each criterion and option. The user rates the importance of one criterion/option over another using a scale provided in the AHP Web Application (e.g., 1 to 9). These comparison values are then stored in matrices.

4.2.3 Normalization and Priority Calculation:

Once the pair-wise comparison matrices are filled, the module normalizes them to ensure consistency and calculates the priority vectors for criteria and options. The normalization involves dividing each element of a row in a matrix by the sum of the elements in that row. The priority vector represents the relative importance of each criterion/option within its respective group.

4.2.4 Consistency Check:

To maintain the accuracy and validity of the AHP results, the module performs a consistency check on the input data. It calculates the consistency index (CI) and compares it with a predefined random index (RI) to ensure that the input data is consistent. If the CI exceeds a certain threshold, the module may prompt the user to reevaluate their pair-wise comparisons for better consistency.

4.2.5 Aggregation of Criteria Priority:

The criteria priorities are aggregated to calculate the overall priorities of the available options. The module multiplies each option's priority vector by the criteria priorities and sums them

up to obtain the overall priorities.

4.2.6 Recommendation Generation:

Based on the calculated overall priorities, the module generates a recommendation for the best option to select. The option with the highest priority is considered the most suitable choice according to the user's preferences and priorities.

4.2.7 Output and Presentation:

The final results, including the priorities of criteria and options and the recommendation, are presented to the user through the AHP Web Application's user interface. The user can visualize the outcomes and make informed decisions based on the AHP analysis.

Conclusion:

The "AHP Implementation" module is the heart of the AHP Web Application, as it performs the essential computations for prioritizing criteria and options based on the user's inputs. It ensures that the AHP process is carried out accurately and efficiently, providing users with valuable insights and recommendations to make well-informed decisions. The module's careful handling of data, normalization, and consistency checks enhances the reliability and effectiveness of the AHP Web Application as a decision support tool in various domains.

4.3 MATH INVOLVED IN AHP :

1. **AHP Preference Scale :**

   1 : Equally preferred

   2 : Equally to moderately preferred

   3 : Moderately preferred

   4 : Moderately to strongly preferred

   5 : strongly preferred

   6 : strongly to very strongly preferred

   7 : very strongly preferred

   8 : very to extremely preferred

   9 : Extremely preferred

2. Let n be the number of criterion and m be the number of options .
3. using the preference scale of AHP create n pair-wise comparison Matrices (it will be square matrix of m rows and m columns A_{mm}) between the options based on criterion and one Factor/criterion Comparison Matrix F_{nn} based on the options.
   - if A is a $m * m$ matrix : A_{mm}
     - $A_{ii} = 1$
     - $A_{ij} = \frac{1}{A_{ji}}$ or $A_{ji} = \frac{1}{A_{ij}}$
   - if F is a $n * n$ matrix : F_{nn}
     - $F_{ii} = 1$
     - $F_{ij} = \frac{1}{F_{ji}}$ or $F_{ji} = \frac{1}{F_{ij}}$
     - where i is number row number and j is column number.
4. Once after creating the Pair-wise comparison Matrix A_{mm} and F_{mm} add the values column and save it as a column total total for each pair-wise comparison Matrices .
   - for a pair-wise comparison matrix A_{mm} with `m` columns it will have m column totals , let us refer that vector as T_m where all the column totals will be present.
5. Now modify all the original Pair-wise comparison Matrices A_{mm} to M_{mm} and F_{mm} to N_{mm} as follows :
   - $M_{ij} = \frac{A_{ij}}{T_k}$ where $k \leq m$
   - $N_{ij} = \frac{F_{ij}}{T_k}$ where $k \leq m$
6. Now we need to calculate the priorities :
   a. Let P_m be the $m + 1$ priority vectors .
   b. To calculate the priorities we need to get the average of each row of Modified Pair-wise matrices i.e M_{mm} and N_{mm}.
7. Now we need to calculate $m + 1$ Weighted sum vectors .
   a. Let $WtSumVec_m$ be the $m + 1$ Weighted sum vectors .
   b. To calculate Weighted sum vectors we need to multiply A_{mm} with P_m

8. Now we need to calculate $m + 1$ Consistency vectors:
    a. Let Cv_m be the $m + 1$ Consistency vectors.
    b. To calculate Consistency vectors we need to divide $WtSumVec_m$ with P_m
9. Now we need to calculate 3 Aspects :
    a. **Lambda : Avg(Cv_m)**
    b. **Consistency Index :** $\frac{Lambda - m}{m - 1}$
    c. **Consistency Ratio :** $\frac{ConsistencyIndex}{RI}$

> 🚧 Note :
>
> 1. Select `RI` Based on the value of m i.e number of options / alternatives .
> 2. Number of Option `m` vs the Ratio Index `RI` :
>
> | m | RI |
> |---|---|
> | 2 | 0.00 |
> | 3 | 0.58 |
> | 4 | 0.90 |
> | 5 | 1.12 |
> | 6 | 1.24 |
> | 7 | 1.32 |
> | 8 | 1.42 |

10. Now create a new matrix Factors vs Options lets call it FM_{mn}
    a. Fill this $m * n$ matrix with the m P_m vectors .
11. To obtain the final Priorities FP_{mn}:
    a. FP_{mn} : Multiply FM_{mn} with the factor priority i.e $m + 1^{th}$ priority vector.
    b. Highest Priority value is the best Decision to go with.

The Analytic Hierarchy Process (AHP)

Design	M1 air	M1 pro	M2 air
M1 air	1.00	3.00	0.11
M1 pro	0.33	1.00	0.13
M2 air	9.00	8.00	1.00
Total	10.33	12.00	1.24

	M1 air	M1 pro	M2 air	Priority
M1 air	0.10	0.25	0.09	0.15
M1 pro	0.03	0.08	0.10	0.07
M2 air	0.87	0.67	0.81	0.78
Total	1.00	1.00	1.00	1.00

wt sum vector
0.4491809074
0.2185322782
2.670099875

consistency vector
3.086
3.025
3.414

Lambda	3.175
CI	0.087
CR	0.151

Features	M1 air	M1 pro	M2 air
M1 air	1.00	0.33	0.10
M1 pro	3.00	1.00	0.17
M2 air	9.00	6.00	1.00
Total	13.00	7.33	1.27

	M1 air	M1 pro	M2 air	Priority
M1 air	0.08	0.04545454545	0.08	0.07
M1 pro	0.23	0.14	0.13	0.17
M2 air	0.69	0.82	0.79	0.77
Total	1.00	1.00	1.00	1.00

wt sum vector
0.1991861939
0.4953379953
2.368053

consistency vector
2.96812919
2.979704797
3.08881421

Lambda	3.012216066
CI	0.00610803280
CR	0.01053109105

Price	M1 air	M1 pro	M2 air
M1 air	1.00	3.00	6.00
M1 pro	0.33	1.00	5.00
M2 air	0.17	0.20	1.00
Total	1.50	4.20	12.00

	M1 air	M1 pro	M2 air	Priority
M1 air	0.67	0.7142857143	0.50	0.63
M1 pro	0.22	0.24	0.42	0.29
M2 air	0.11	0.05	0.08	0.08
Total	1.00	1.00	1.00	1.00

wt sum vector
1.988095238
0.9047619048
0.2436507937

consistency vector
3.170886076
3.095022624
3.019672131

Lambda	3.095193611
CI	0.04759680526
CR	0.08206345734

Factor	Features	Design	Price
Features	1.00	9.00	0.1428571429
Design	0.11	1.00	0.1111111111
Price	7.00	9.00	1.00
Total	8.11	19.00	1.253968254

	Features	Design	Price	Priority
Features	0.12	0.4736842105	0.11	0.24
Design	0.01	0.05	0.09	0.05
Price	0.86	0.47	0.80	0.71
Total	1.00	1.00	1.00	1.00

wt sum vector
0.8034056878
0.1570186086
2.834959342

consistency vector
3.390393662
3.040289804
3.985105646

Lambda	3.471929704
CI	0.2359648519
CR	0.4068359516

	Features	Design	Price	Priority
M1 air	0.15	0.07	0.63	0.4839865654
M1 pro	0.07	0.17	0.29	0.2336623398
M2 air	0.78	0.77	0.08	0.2823510948

n	RI
2.00	0.00
3.00	0.58
4.00	0.90
5.00	1.12
6.00	1.24
7.00	1.32
8.00	1.41

5. System Study

In the system study phase of the AHP Web Application development, various aspects of the project are analyzed to determine its feasibility and practicality. This phase aims to assess the project's viability and identify potential challenges that may arise during implementation. The system study involves conducting a feasibility study that comprises three key components: **Technical Feasibility, Operational Feasibility, and Economic Feasibility.**

5.1 Feasibility Study

5.1.1 Technical Feasibility:

During the technical feasibility assessment, the development team evaluates whether the required technologies and resources are readily available to build the AHP Web Application. It involves analyzing the compatibility of the selected programming languages (Python), libraries (Streamlit, Pandas, NumPy), and other tools with the application's functionalities. Additionally, the team assesses the capability of the web hosting infrastructure to handle potential user traffic and data storage requirements. The technical feasibility study confirms that the necessary technical components and skills are present to create a robust and efficient AHP Web Application.

5.1.2 Operational Feasibility:

The operational feasibility analysis focuses on the application's ability to fit seamlessly into the existing business operations or decision-making processes. It involves considering the ease of use and user-friendliness of the web application's interface. The AHP Web Application is designed to be intuitive, guiding users through a step-by-step process to define their goals, criteria, and options. The operational feasibility study aims to ensure that the AHP Web Application aligns with users' expectations and requirements, making it a valuable tool for decision-makers across various domains.

5.1.3 Economic Feasibility:

The economic feasibility assessment evaluates the financial viability of the AHP Web Application project. It involves estimating the overall project costs, including development, hosting, and maintenance expenses. The team considers the potential benefits the application can bring in terms of improved decision-making and time-saving.

6. GRAPHICAL USER INTERFACE

The Graphical User Interface (GUI) is a critical component of the AHP Web Application, as it is the primary interaction point between users and the system. The GUI is designed to be user-friendly, intuitive, and visually appealing, facilitating a seamless user experience throughout the decision-making process. It comprises two key aspects: Input Design and Output Design.

6.1 Input Design

The Input Design of the AHP Web Application is focused on providing users with an easy and structured way to input their preferences, criteria, and options for the decision-making process. The design aims to guide users through a step-by-step process to ensure completeness and accuracy in defining their goals and priorities. The following elements are included in the Input Design:

6.1.1 Goal Definition:

The GUI begins by presenting a clear and prominent field where users can define their decision-making goal. Users are encouraged to provide a concise and descriptive goal statement that reflects the purpose of their decision.

6.1.2 Criteria and Option Inputs:

After defining the goal, users are prompted to enter the criteria and options relevant to their decision. The GUI allows dynamic addition of criteria and options based on user preferences. Users can easily add or remove items using interactive buttons.

6.1.3 Pair-wise Comparison Matrices:

The AHP algorithm relies on pair-wise comparisons to determine the relative importance of criteria and options. The GUI presents users with matrices where they can provide their preferences for each pair of criteria and options. A consistent and user-friendly rating scale (e.g., 1 to 9) is used to facilitate the comparison process.

6.1.4 Feedback and Validation:

The GUI provides real-time feedback to users during the input process. It validates user inputs to prevent errors and inconsistencies. If necessary, informative messages guide users to correct any incomplete or inconsistent data.

6.2 OUTPUT DESIGN

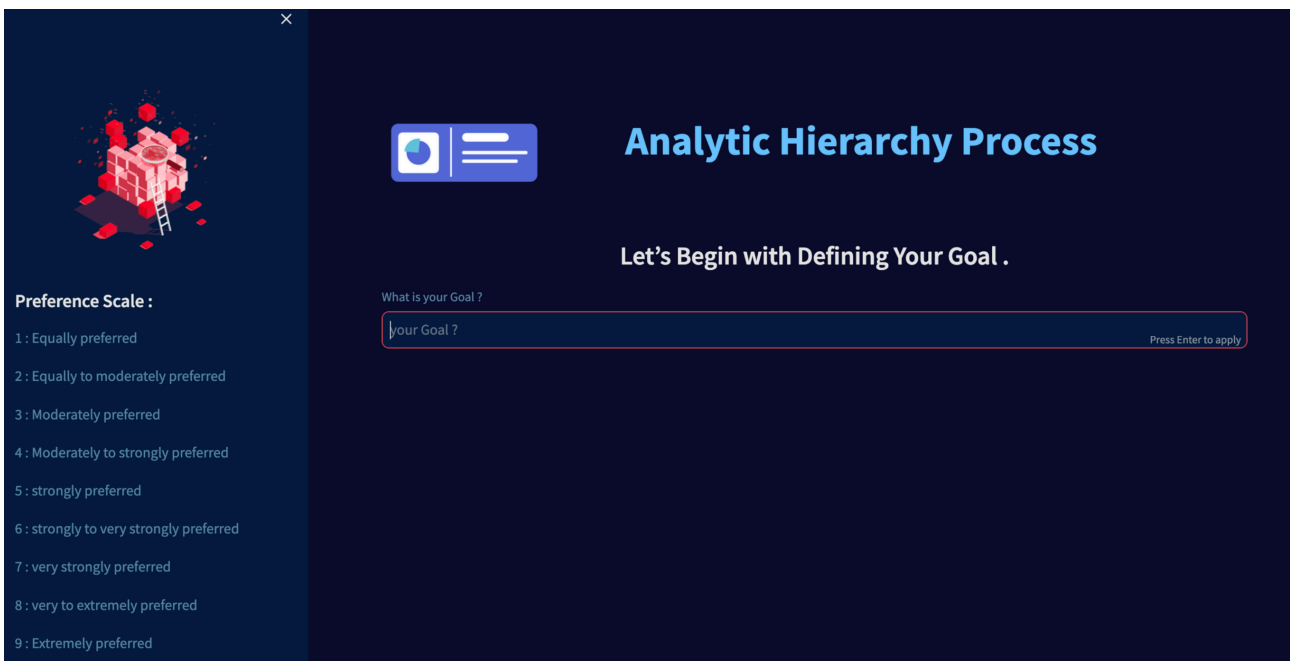
The Output Design of the AHP Web Application focuses on presenting the results of the decision-making process in a clear and insightful manner. It ensures that users can easily interpret the prioritized criteria, options, and the final recommendation. The following elements are included in the Output Design:

6.2.1 Prioritized Criteria and Options:
The GUI displays the priorities of criteria and options after the AHP algorithm processes the user inputs. It presents the results in a well-organized table or graphical format, allowing users to comprehend the relative importance of each criterion and option.

6.2.2 Recommendation and Visualization:
Based on the calculated priorities, the GUI generates a final recommendation for the best option to select. The recommended option is clearly highlighted, and users can visualize the decision-making outcome with intuitive charts or graphs.

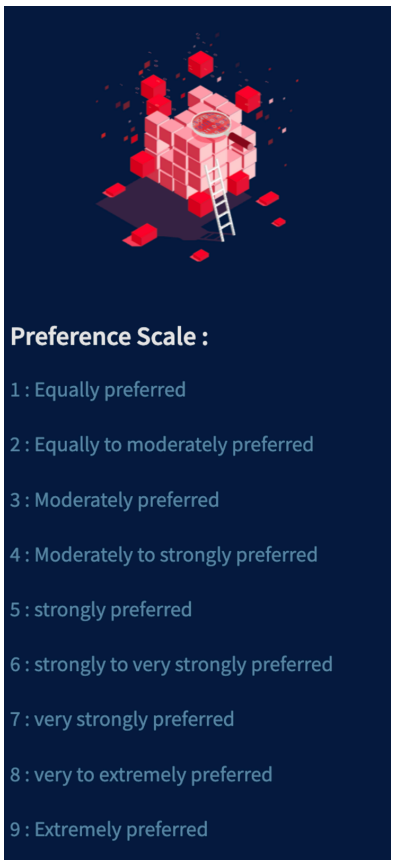
6.2.3 Consistency Check and Alerts:
In case the user inputs do not meet the required consistency criteria, the GUI provides relevant alerts and messages. It encourages users to review their pair-wise comparisons and make adjustments for better results.

6.2.4 User Interaction:
The GUI allows users to interact with the output, such as exploring different scenarios by modifying criteria or changing the input preferences. Users can see how the final recommendation changes based on their adjustments.

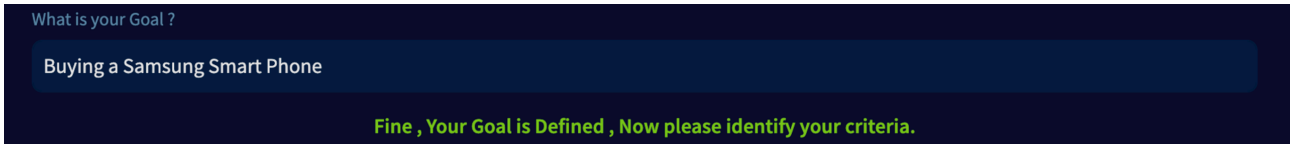


6.3 SCREEN SHOTS :

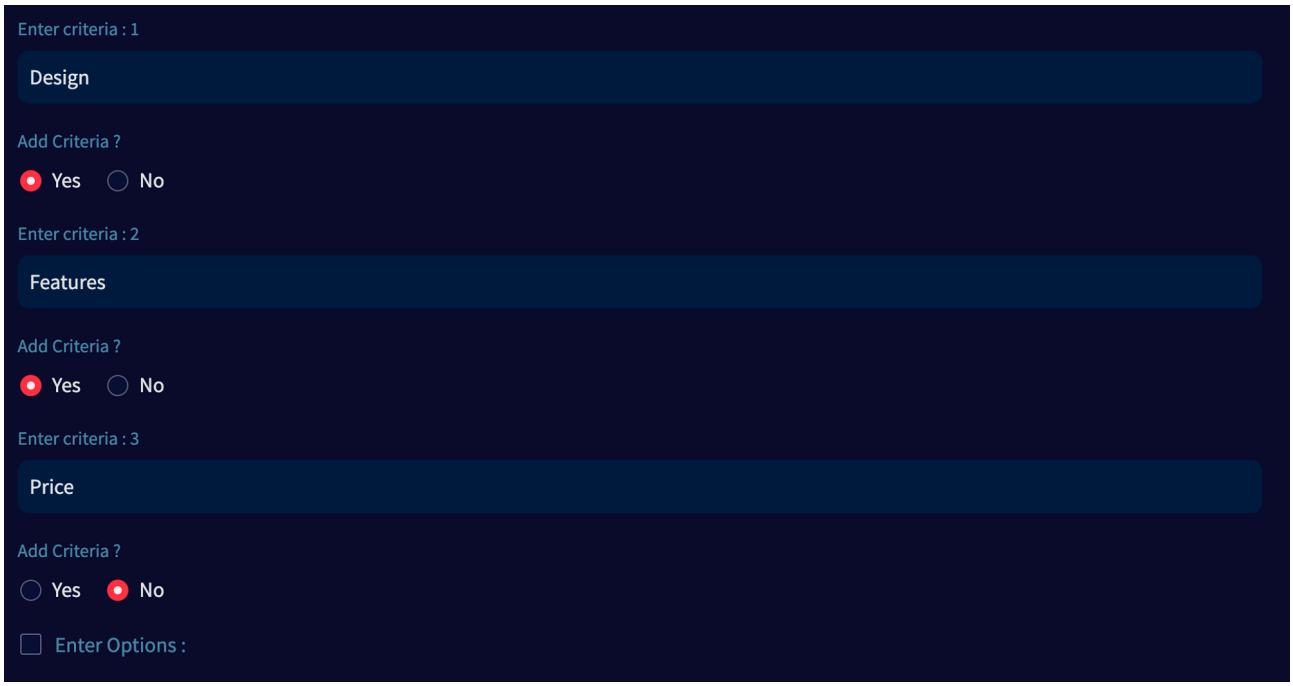
6.3.1 AHP User Interface :



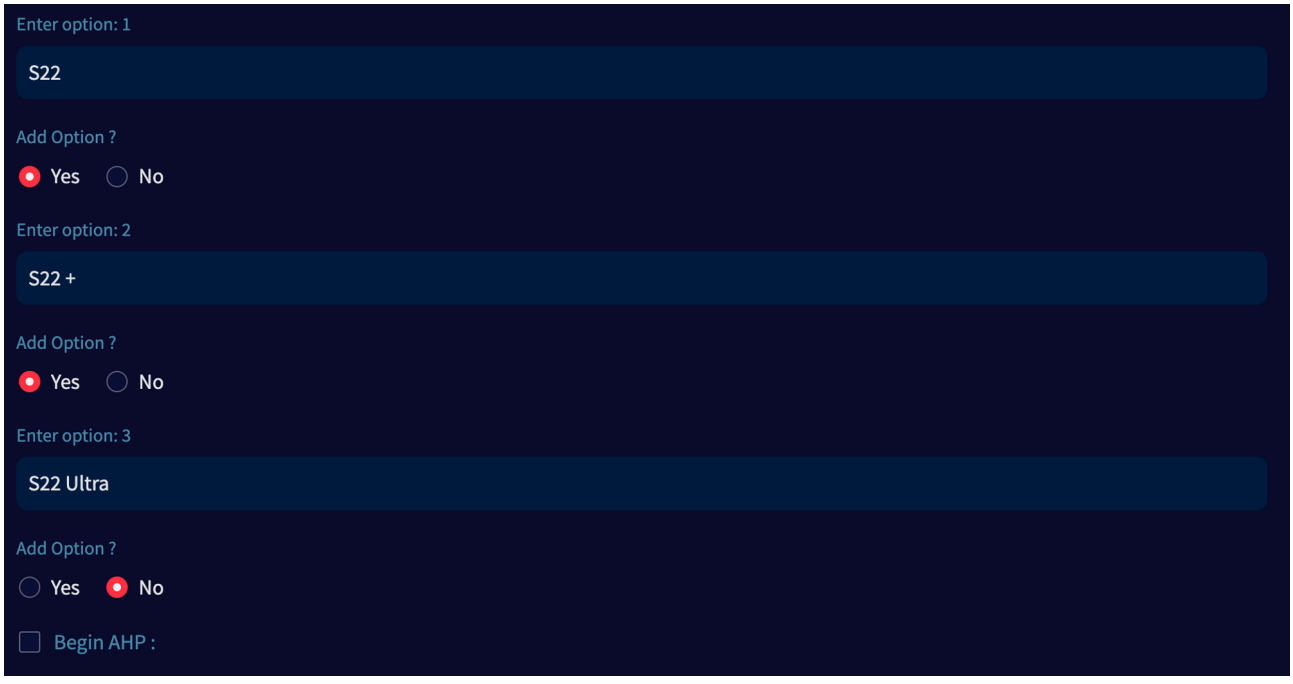
6.3.2 AHP Preference Scale :



6.3.3 Defining Goal for AHP

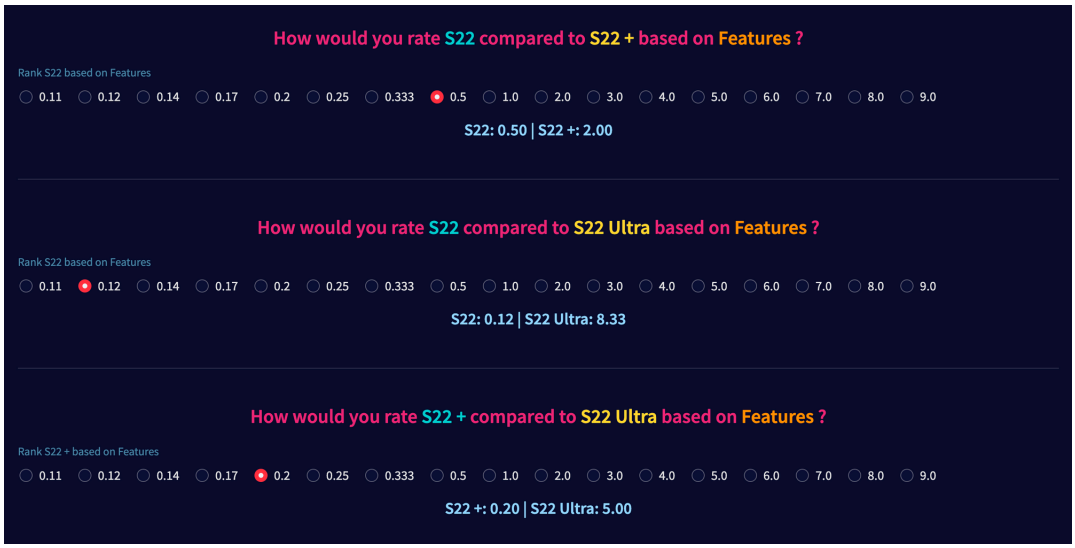


6.3.4 Defining Criteria



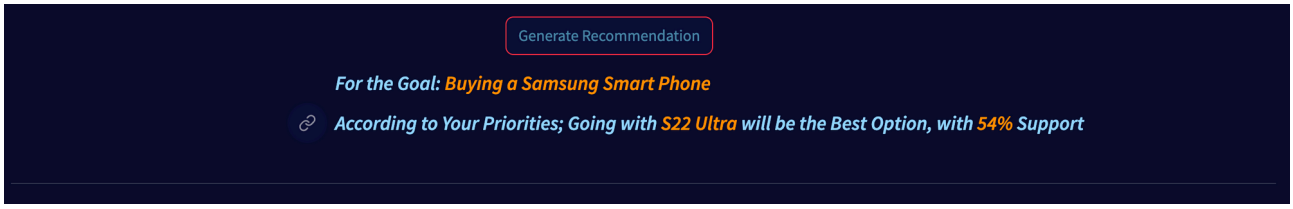
6.3.5 Defining Options

6.3.6 AHP Quesioning :



6.3.6.1 comparing options based on criterion :

6.3.6.2 criteria comparision :



6.3.7 Generate Recommendation

7. TESTING

Testing is a crucial phase in the development of the AHP Web Application. It involves evaluating the application's functionalities, features, and overall performance to ensure its correctness, reliability, and robustness. The testing phase is divided into several categories, including System Testing, Unit Testing, Integration Testing, and Acceptance Testing. Each type of testing serves a specific purpose and helps in identifying and resolving potential issues.

7.1 System Testing

The System Testing is a comprehensive testing process that evaluates the entire AHP Web Application as a complete system. It aims to verify whether all the components and modules of the application work together as expected and meet the specified requirements. The System Testing phase ensures that the application functions as a cohesive unit, delivering the desired results to users.

7.2 Types of Testing

7.2.1 Unit Testing

Unit Testing focuses on testing individual units or components of the AHP Web Application in isolation. Each unit, such as functions, methods, or classes, is tested to ensure its correctness and proper functioning. Unit Testing is essential in identifying and fixing bugs at an early stage, which contributes to the overall reliability and maintainability of the application.

7.2.2 Integration Testing

Integration Testing evaluates the interactions and interfaces between different units or components of the AHP Web Application. It verifies the smooth integration of these units to guarantee that they work cohesively when combined. Integration Testing helps uncover any communication issues or data inconsistencies between the integrated components.

7.2.3 Functional Testing

Functional Testing is centered on evaluating the application's functionalities and features. It ensures that the AHP Web Application performs all the expected tasks accurately and adheres to the specified requirements. Functional Testing includes various scenarios and user interactions to validate the application's correctness and usability.

7.2.4 System Testing

System Testing, as mentioned earlier, focuses on the overall system behavior. It checks the application's compatibility with different browsers, operating systems, and hardware configurations. System Testing validates the application's responsiveness, performance, and stability under different conditions to ensure its effectiveness in real-world scenarios.

7.2.5 White Box Testing

White Box Testing, also known as structural testing or code-based testing, involves examining the internal structure of the AHP Web Application's code. Testers have access to the application's source code and use it to design test cases that target specific code paths, conditions, and data structures. White Box Testing helps identify issues related to code logic and execution.

7.2.6 Black Box Testing

Black Box Testing, also known as functional testing, focuses on evaluating the application's functionalities from an end-user perspective. Testers are not concerned with the application's internal code or structure. Instead, they design test cases based on the application's requirements and expected behavior. Black Box Testing validates the application's correctness, input validation, and error handling.

7.3 Unit Testing

In the context of the AHP Web Application, Unit Testing involves testing individual functions and modules that constitute the application. Each function is tested independently to ensure it performs the intended tasks accurately. Unit Testing helps identify and resolve bugs and ensures that each function meets its specified requirements.

7.4 Integration Testing

Integration Testing for the AHP Web Application verifies the seamless integration of different modules and components. It ensures that data flows smoothly between various functionalities, and the application works cohesively as a whole. Integration Testing helps detect issues related to data transfer, communication, and interoperability between different parts of the application.

7.5 Acceptance Testing

Acceptance Testing is the final phase of testing and involves evaluating the AHP Web Application's readiness for deployment. It is conducted to determine if the application meets the end-users' expectations and fulfills the specified requirements. Acceptance Testing is typically performed by end-users or stakeholders to validate the application's suitability for the intended use.

Conclusion:

The testing phase is a critical part of the development process for the AHP Web Application. It ensures that the application performs as expected, meets the specified requirements, and delivers a seamless and reliable user experience. System Testing, Unit Testing, Integration Testing, Functional Testing, White Box Testing, and Black Box Testing collectively contribute to the overall quality and success of the AHP Web Application. By detecting and resolving issues at each level of testing, the development team ensures that the final application is robust, accurate, and capable of providing valuable decision-making support to users.

8. CONCLUSION

The AHP Web Application presents a powerful and user-friendly tool for decision-making using the Analytic Hierarchy Process (AHP). Throughout the development of this application, various modules and functionalities were carefully designed and implemented to ensure a seamless and efficient user experience. The project has successfully achieved its objectives, providing users with a structured and intuitive platform to prioritize multiple criteria and options and arrive at the best decision.

The Graphical User Interface (GUI) plays a pivotal role in the AHP Web Application, facilitating a user-friendly experience. The Input Design enables users to define their decision-making goals, input criteria and options, and perform pair-wise comparisons easily. The real-time feedback and validation mechanisms ensure that users provide accurate and consistent inputs, further enhancing the reliability of the decision-making process.

The Output Design of the AHP Web Application effectively presents the results of the AHP algorithm in a clear and insightful manner. The prioritized criteria and options are displayed in an organized format, allowing users to grasp the relative importance of each factor quickly. The final recommendation is highlighted, enabling users to make informed choices based on their priorities. Additionally, the consistency check and alerts provide valuable feedback to users, ensuring that their decisions adhere to the required criteria.

The testing phase, including System Testing, Unit Testing, Integration Testing, Functional Testing, White Box Testing, and Black Box Testing, was meticulously carried out to ensure the application's correctness and robustness. System Testing verified the application's complete functionality and performance, while Unit and Integration Testing focused on individual components' accuracy and seamless integration. Functional Testing ensured that the application meets all requirements, and White Box and Black Box Testing scrutinized the internal and external behaviors of the application. Acceptance Testing provided the final validation by stakeholders and end-users, confirming the application's suitability for real-world use.

The AHP Web Application demonstrates its significance and potential in various decision-making scenarios, including business, personal, and academic contexts.

Its versatility and ease of use make it a valuable asset to users seeking to prioritize and select the best options based on their preferences. The systematic approach of the Analytic Hierarchy Process ensures that users make well-informed and rational decisions, avoiding biases and subjective judgments.

Moreover, the AHP Web Application can be further enhanced with future updates and improvements. Additional features, such as sensitivity analysis, advanced visualization options, and multi-criteria decision-making, could expand the application's capabilities and cater to a broader range of decision-making scenarios.

Overall, the AHP Web Application stands as a testament to effective software development practices and the application of the Analytic Hierarchy Process. It empowers users with a structured and efficient decision-making approach, enabling them to make more informed and optimal choices. As technology continues to advance, this web application holds the potential to become an essential tool for decision-makers across various domains. With its user-friendly interface, precise analysis, and reliable results, the AHP Web Application promises to be a valuable asset in the pursuit of effective decision-making.

9 REFERENCES

1. Saaty, T. L. (1980). The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. New York: McGraw-Hill. This is the seminal work by Thomas L. Saaty, the creator of the Analytic Hierarchy Process. It provides a comprehensive overview of the AHP methodology and its applications.

2. Vaidya, O. S., & Kumar, S. (2006). Analytic Hierarchy Process: An Overview of Applications. European Journal of Operational Research, 169(1), 1-29. This paper presents an extensive review of various applications of the AHP across different domains, including business, engineering, and healthcare.

3. Opricovic, S., & Tzeng, G. H. (2004). Compromise Solution by MCDM Methods: A Comparative Analysis of VIKOR and TOPSIS. European Journal of Operational Research, 156(2), 445-455. This paper discusses multi-criteria decision-making methods, including VIKOR, which is related to the AHP and can be useful for further enhancing your web application's capabilities.

4. Tzeng, G. H., & Huang, J. J. (2011). Multiple Attribute Decision Making: Methods and Applications. Boca Raton: CRC Press. This book provides a comprehensive overview of various multiple attribute decision-making methods, including the AHP, and offers practical examples and applications.

5. Molina, A., & Morales, D. (2008). AHP and Goal Programming for Planning and Scheduling in a Cellular Manufacturing System. Omega, 36(6), 961-976. This paper showcases a specific application of the AHP in manufacturing planning and scheduling, illustrating how the methodology can be used in real-world scenarios.

10.1 APPENDIX - A

Sample code : app.py

```
import pandas as pd
import numpy as np
import streamlit as st
import time
import json
import requests
from streamlit_lottie import st_lottie
st.set_page_config(page_title="AHP", page_icon='😉',
layout="wide",initial_sidebar_state='auto')
# Using the st_lottie function to load and display the animation
def load_lottie_url(url:str):
    r=requests.get(url)
    if r.status_code!=200:
        st.warning('No image')
        return None
    return r.json()

def load_lottie_file(filepath:str):
    with open(filepath,"r") as f:
        return json.load(f)
lottie_file1 =load_lottie_file('./assets/f1.json')
lottie_file2 =load_lottie_file('./assets/f2.json')
# Hide the "Made with Streamlit" footer
hide_streamlit_style="""
    <style>
    #MainMenu{visibility:hidden;}
    footer{visibility:hidden;}
    h1{
        color: #66C4FF;
        }
    h4{
        # color: #FF5733;
        color:   #E04078;
        text-align: center;
        }
    h5{
        color: #A3D7FF;
        }
    span {
        # color : #A3D7FF;
        font-size: inherit;
        font-weight: inherit;
    }
    h6{
        color: #7AC917;
        # color:  #A3D7FF;
        }
    p{

        color:  #5A88A6;
        # color: skyblue;
        }
    .option1 {
        color:#00CED1;
    }
    .option2 {
        color: #FFDB58;
    }
    .factors {
        color: #FF9800;
    }
    </style>


    """
st.markdown(hide_streamlit_style,unsafe_allow_html=True)
```

```python
c1 , c2  = st.columns([0.75,2])
with c1 :
    st_lottie(lottie_file1,speed=0.5,reverse=False,height=120,width=180)
with c2 :
    st.title("Analytic Hierarchy Process",anchor=False)
# Gloabal Variables
add_Criteria = ['Yes','No']
add_Options = ['Yes','No']
rating = [0.11,0.12,0.14,0.17,0.20,0.25,0.333,0.50,1.00,2.00,3.00,4.00,5.00,6.00,7.00,8.00,9.00]
n=1
m=1
status =False
Factors = []
Options = []
def subheadingtext(text:str):
    message = []
    response = st.empty()
    tokens = list(text)
    for i in tokens:
        message.append(i)
        result = "".join(message)
        response.markdown(f'##### {result} ',unsafe_allow_html=True)
        time.sleep(0.012)
# st.divider()
with st.sidebar:
    st_lottie(lottie_file2,speed=0.5,reverse=False,height=180,width=280)
    st.header("Preference Scale :")
    st.write("""
            1 : Equally preferred\n
            2 : Equally to moderately preferred\n
            3 : Moderately preferred\n
            4 : Moderately to strongly preferred\n
            5 : strongly preferred\n
            6 : strongly to very strongly preferred\n
            7 : very strongly preferred\n
            8 : very to extremely preferred\n
            9 : Extremely preferred\n
        """)
st.markdown(f"<h3 style='text-align: center;'>Let’s Begin with Defining Your Goal .</h3>",
unsafe_allow_html=True)
```

```
goal = st.text_input("What is your Goal ?",placeholder="your Goal ?")
if goal!="":
    st.markdown(f"<h6 style='text-align: center;'> Fine , Your Goal is Defined , Now please identify
your criteria.</h6>", unsafe_allow_html=True)
    criteria = st.text_input(f"Enter criteria : {n}",key="criteria"+str(n),placeholder="Factor
"+str(n))
    Factors.append(criteria)
    if criteria !="":
        while n:
            choose = st.radio('Add Criteria
?',add_Criteria,index=1,key="choose"+str(n),horizontal=True)
            if choose=='Yes':
                n+=1
                for i in range (n,n+1):
                    criteriaAdd = st.text_input(f"Enter criteria :
{i}",key="criteriaAdd"+str(i),placeholder="Factor "+str(i))
                    Factors.append(criteriaAdd)
            else :
                # st.warning(criteria)
                opt=st.checkbox("Enter Options :")
                if opt:
                    status = True
                # st.info(choose)
                # status = True
                break
if status:
    #option priority :
    option = st.text_input(f"Enter option: {m}",key="optionAdd"+str(m))
    Options.append(option)
    if (option!=""):
        while m:
            addOption = st.radio('Add Option
?',add_Options,index=1,key="addOption"+str(m),horizontal=True)
            if addOption=='Yes':
                m+=1
                for i in range(m,m+1):
                    option = st.text_input(f"Enter option: {i}",key="optionAdd"+str(i))
                    Options.append(option)
            else:
                break
```

```
if (((len(Options)) and (len(Factors)))):
    process = st.checkbox("Begin AHP :")
    try:
        if process:
            # AHP(n,m)
            final_df = pd.DataFrame(data = [], index = Options )
            RI = [0,0,0,0.58,0.90,1.12,1.24,1.32,1.42]
            count = 0
            st.divider()
            for i in range (n):
                data = {Factors[i] : Options}
                for j in range (m):
                    l = [1.]*m
                    data[Options[j]] = l
                df = pd.DataFrame(data).set_index(Factors[i])
                for j in range (m-1):
                    for k  in range (j+1,m):
                        st.markdown(f'<h4>How would you rate <span class="option1">{Options[j]}</span>
compared to <span class="option2">{Options[k]}</span> based on <span class="factors">{Factors[i]}
</span> ?</h4>', unsafe_allow_html=True)

                        df.iloc[j][k] = st.radio(f"Rank {Options[j]} based on
{Factors[i]}",options=rating,index=8,horizontal=True,key="radio"+str(count))
                        count+=1
                        df.iloc[k][j] = 1 / df.iloc[j][k]
                        st.markdown(f"<h5 style='text-align: center;'>{Options[j]}: {(df.iloc[j]
[k]):.2f} | {Options[k]}: {(df.iloc[k][j]):.2f}</h5>", unsafe_allow_html=True)
                        st.divider()
                df_org = df.copy()
                df = df.div(df.sum())
                df['Priorities'] = np.mean(df,axis=1)
                dot_product = np.dot(df_org,df['Priorities'])
                consistency_vector = dot_product / df ['Priorities']
                Lambda = consistency_vector.mean()
                consistency_index = (Lambda - m)/(m-1)
                if(df_org.values.sum()%3==0):
                    st.write("update priorities")
                    exit(0)
                consistency_ratio = consistency_index / RI[m]
                if((consistency_ratio > 0.1)):
                    st.error(f"Consistency Ratio {consistency_ratio:.2f} > 0.1")
                    exit(0)
                final_df[Factors[i]] = df ['Priorities']
            # FP (Factors Priority)
            FP = pd.DataFrame(data = np.ones((n,n)),index = Factors)
            # st.dataframe(FP)
            # st.divider()
            count = 0 # for keeping the key unique
            for i in range (n-1):
                for j in range (i+1,n):
                    st.markdown(f'<h4> <span class="option2">{Factors[i]}</span> vs <span
class="factors">{Factors[j]}</span></h4>', unsafe_allow_html=True)
                    FP.iloc[i][j] = st.radio(f"Evaluate {Factors[i]} with respect to
{Factors[j]}",options=rating,index=8,horizontal=True,key="radio "+str(count))
                    count+=1
                    FP.iloc[j][i] = (1)/(FP.iloc[i][j])
                    st.markdown(f"<h5 style='text-align: center;'>{Factors[i]}: {(FP.iloc[i][j]):.2f} |
{Factors[j]}: {(FP.iloc[j][i]):.2f}</h5>", unsafe_allow_html=True)

            FP_org = FP.copy()
            FP = FP.div(FP.sum())
            FP['Priorities'] = np.mean(FP,axis=1)
            dot_product = np.dot(final_df,FP['Priorities'])
            # dot_product
            final_df['Priorities'] = dot_product
            # final_df
            message1 = str(final_df.loc[final_df['Priorities'] ==
final_df['Priorities'].max()].index[0])+" "
            message2 = str(int(final_df['Priorities'].max()*100))
            # st.divider()
            exp =False
            # with st.expander("See Result :"):
            col1 , col2  = st.columns([2.5,4])
            col3 , col4 = st.columns([1,3])
            st.divider()
            with col2:
                if st.button("Generate Recommendation",help="submit",key="1"):
                        with col4:
                            subheadingtext(f'*For the Goal:  <span class="factors">{goal}*<span>')
                            subheadingtext(f'*According to Your Priorities; Going with <span
class="factors">{message1}</span> will be the Best Option, with <span class="factors">{message2}%
</span> Support*')
    except IndexError:
        st.error("please Answer the input fields .")
```

10.2 APPENDIX B

Software and Hardware Requirements

Software Requirements:

1. Python: The latest version of Python should be installed on the system. The web application is built using Python, so it is essential to have the Python interpreter available.

2. Streamlit: Install the Streamlit library using pip, which is a Python package manager. Streamlit is used to create the web application interface and functionalities.

3. NumPy: Ensure that NumPy, the numerical computing library, is installed. It is used for mathematical calculations and data manipulation in the AHP Web App.

4. Pandas: Install Pandas, a data manipulation library, as it is used to manage and process data in the web application.

5. Requests: The Requests library is required for handling HTTP requests and fetching external resources, such as animations, in the web app.

6. Lottie: Install Lottie, a library for embedding animations in web applications using JSON files. Lottie is used to display animations in the AHP Web App.

7. Web Browser: A modern web browser, such as Google Chrome, Mozilla Firefox, or Microsoft Edge, is needed to access and interact with the AHP Web Application.

Hardware Requirements:

1. Processor: A computer or device with at least a dual-core processor is recommended. The web application may perform complex calculations, so a faster processor will provide a smoother user experience.

2. RAM: The system should have sufficient RAM to handle the web application's processes. At least 4GB of RAM is recommended for optimal performance.

3. Storage: The AHP Web Application itself is lightweight, but the system should have enough storage space to accommodate Python and the necessary libraries.

4. Internet Connection: Since the web application may require fetching external resources or accessing online data, an active internet connection is preferable for the best user experience.

5. Display: A monitor or screen with a resolution of at least 1280x720 is recommended to ensure that the web application's interface and visualizations are displayed correctly.

6. Input Devices: The system should have standard input devices such as a keyboard and mouse to interact with the web application effectively.

10.3 APPENDIX - C

Technology Used

1. **Python:** Python is the primary programming language used to develop the AHP Web Application. Python is known for its simplicity, readability, and vast library support, making it a popular choice for web development. It allows developers to implement complex algorithms, handle data manipulation, and build interactive web interfaces with ease.

2. **Streamlit:** Streamlit is a Python library specifically designed for creating web applications with minimal effort. It enables developers to transform data scripts into interactive web apps without the need for extensive front-end development. Streamlit provides various widgets and interactive components that allow users to input data and visualize results effectively.

3. **NumPy:** NumPy is a fundamental Python library for scientific computing. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy is essential for implementing mathematical calculations and algorithms, which are integral to the AHP methodology.

4. **Pandas:** Pandas is another crucial Python library used in data manipulation and analysis. It provides data structures, such as DataFrame and Series, that allow developers to handle and analyze data effectively. In the AHP Web Application, Pandas is likely used to manage and process the user input data, perform calculations, and generate outputs.

5. **JSON:** JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. In the AHP Web Application, JSON is used to load and store animations and other resources.

6. **Requests:** Requests is a Python library that allows the AHP Web Application to make HTTP requests, such as fetching animations or other external resources from URLs. It is essential for handling external data and resources.

7. **HTML/CSS:** Although not explicitly mentioned in the provided code, it is likely that HTML and CSS are used for creating the structure and styling of the web pages in the AHP Web Application. HTML is used for creating the web page's content and layout, while CSS is used for styling and visual presentation.

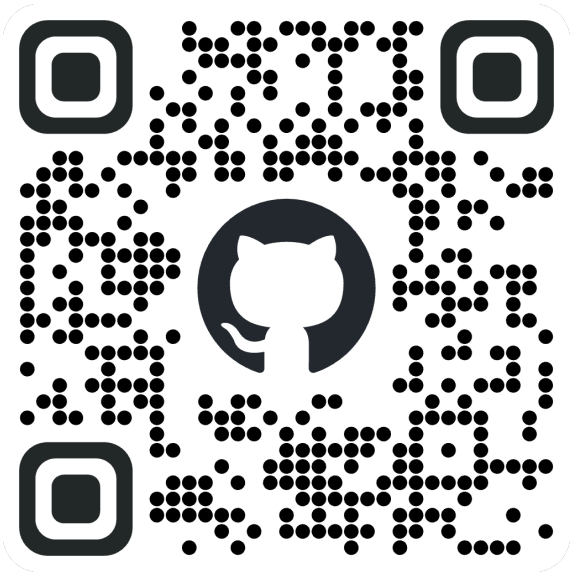
8. **Lottie:** Lottie is a library that allows developers to embed animations in a web application using JSON files. In the AHP Web Application, Lottie is used to display animations, enhancing the user interface and experience.

Project Structure :

11.1 Project Structure

QR Codes:

12.1 Code GitHub Repository :



https://github.com/Mohammed-Khubaib/AHP

12.2 Documents :

https://drive.google.com/drive/folders/1ZiB8IIPNJYO51NgRk-aA0k-Clts87b_7?usp=drive_link