CHRIST

(DEEMED TO BE UNIVERSITY)

BANGALORE · INDIA

DEPARTMENT OF

ARTIFICIAL INTELLIGENCE, MACHINE LEARNING AND DATA SCIENCE

Mini-Project Report

Data Structures

(CSE332P)

B. Tech in Computer Science with spl. in AIML Degree – 3 BTCSAIML C

School of Engineering and Technology,

CHRIST (Deemed to be University),

Kumbalagodu, Bengaluru-560 074

September 2025

Certificate

This is to certify that Alicia Theresa Pereira (2462028), Joann Binny (2462088) and Stacy Anna Dsouza (2462156) has successfully completed the Mini Project work for Data Structures – CSE332P in partial fulfillment for the award of Bachelor of Technology during the academic year 2024-2025.

FACULTY- IN CHARGE

Athulya S

Names	: Alicia Theresa Pereira	(2462028)
	: Joann Binny	(2462088)
	: Stacy Anna Dsouza	(2462156)

INDEX

Title of the Project

TRAVEL DESTINATION RECOMMENDATION SYSTEM USING BINARY DECISION TREE

Objective and Relevance:

1. Objective:

The primary objective of this project is to develop a **Travel Recommendation System** using the C programming language. The system utilizes a **binary decision tree** to interactively guide users through a series of personalized questions and preferences—such as climate choice, travel companions, and activity interests—to ultimately provide destination recommendations tailored to the user's input.

Key goals include:

- Implementing a **tree-based structure** to simulate decision-making.
- Practicing **dynamic memory allocation** and **recursive function design**.
- Providing an engaging **command-line interface** that mimics a real-world recommendation engine.

2. Relevance

This project is relevant both in terms of **educational value** and **practical application** in the fields of software development and data-driven systems.

Educational Relevance

- Demonstrates the application of **binary trees**, a core data structure, in a real-world problem.
- Reinforces the use of **functions, structures, and pointers** in C.
- Encourages problem-solving through the implementation of **recursive logic** and **user-driven flows**.

Practical Relevance

- Mimics the logic used in **expert systems** and **recommendation engines**, which are widely used in industries like travel, e-commerce, and entertainment.
- Provides a basic structure that can be expanded into more complex systems, including:
    - GUI-based travel planners
    - Web or mobile applications
    - AI-assisted decision-making tools

Skill Development

- Strengthens understanding of:
    - Input validation
    - Memory management
    - Modular and maintainable code design

- Enhances capability to map logical decision processes into software algorithms.

Concept Applied:

The main concept applied in this project is the use of a Binary Tree data structure to build a travel destination recommendation system.

What is a Binary Tree?

A Binary Tree is a hierarchical data structure where each node has at most two children — referred to as the left and right child. In this project, the binary tree is used to represent a structured set of decision-making questions, with each level guiding the user closer to a final travel destination.

Why Use a Binary Tree in This Project?

- Binary Decision-Making
  Each question asked to the user has exactly two options (e.g., Hot or Cool climate), which fits the left/right branching structure of a binary tree.

- Hierarchical Flow of Questions
  Each level of the tree represents a different category of user preference:

  - Level 1 → Climate (Hot / Cool)
  - Level 2 → Trip Type (Solo / Family)
  - Level 3 → Activity (Adventure / Relaxation / Other)
  - Level 4 → Destination (Leaf Node)

- Efficient Traversal
  The tree allows quick navigation based on user responses, moving step-by-step through the tree until a suitable destination is found.

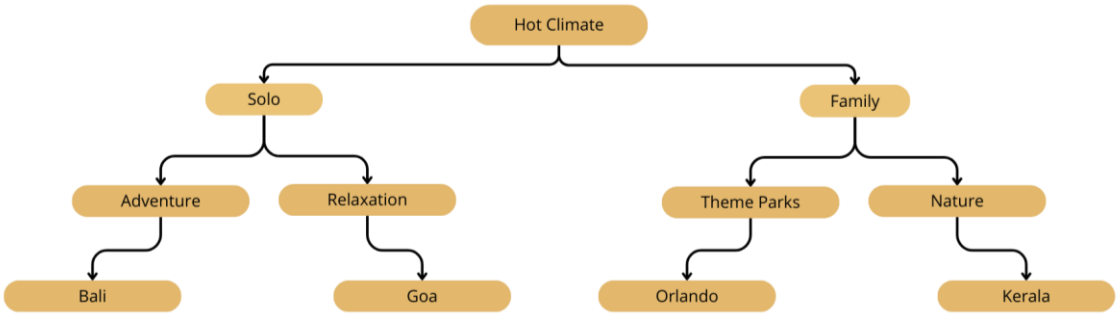
- Easy to Implement in C
  The binary tree structure is simple to implement using structs and pointers in the C programming language, making it ideal for student-level projects.

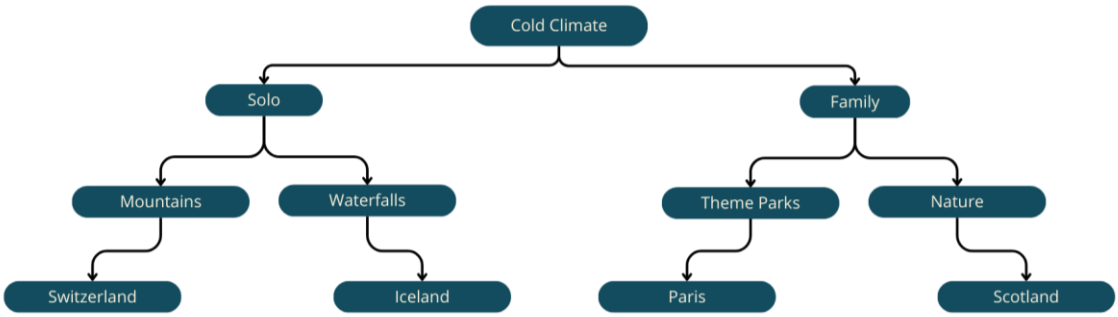
- Practical Real-Life Application
  This project simulates how real-world recommendation systems function by using user input to make logical decisions and return relevant suggestions.

Algorithm:

- Start program.
- Display welcome message (catchphrase + intro lines).
- Ask the first question: Hot (1) or Cool (2) climate.
- If 1 → go to the Hot branch.
- If 2 → go to the Cool branch.
- Ask the second question: Solo/Family (1) or Friends/Work (2).
- Move to the corresponding branch.
- Ask third question:
    - If Solo or Family → Ask: Solo (1) or Family (2).
    - If Friends or Work → Ask: Friends (1) or Work (2).
- Based on group type, ask preference:
    - Adventure (1) or Relaxation (2),
    - or other relevant options depending on previous choices.
- Move deeper into the tree according to user choices.
- Reach a leaf node (final destination recommendation).
- Print travel recommendation with proper formatting.
- Display outro or thank-you message.
- End program.

TREE VISUALIZATION:





Pseudocode of the implementation:

```
START
Display "Welcome to the Travel Recommendation System"
CALL ask(climate)

FUNCTION ask(node)
    IF node is NULL THEN
        RETURN
    ENDIF

    DISPLAY node.text

    IF node.left is NULL AND node.right is NULL THEN
        // It's a recommendation (leaf node)
        DISPLAY "==================================="
        DISPLAY node.text
        DISPLAY "==================================="
        RETURN
    ENDIF

    PROMPT user to enter choice (1 or 2)
    READ choice

    IF choice == 1 THEN
        CALL ask(node.left)
    ELSE IF choice == 2 THEN
        CALL ask(node.right)
    ELSE
        DISPLAY "Invalid choice!"
    ENDIF
END FUNCTION

FUNCTION makeNode(text)
    CREATE newNode
    SET newNode.text = text
    SET newNode.left = NULL
    SET newNode.right = NULL
    RETURN newNode
END FUNCTION
// In main:
CREATE root node for "Climate" question using makeNode
BUILD the binary decision tree with all question and recommendation
nodes
LINK all nodes accordingly (left/right based on options)
CALL ask(root node)
END
```

Program Code:

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct node {
    char text[500];
    struct node *left;
    struct node *right;
} node;

node* makeNode(const char* str) {
    node* temp = (node*)malloc(sizeof(node));
    if (!temp) {
        printf("Memory error!\n");
        exit(1);
    }
    snprintf(temp->text, sizeof(temp->text), "%s", str);
    temp->left = NULL;
    temp->right = NULL;
    return temp;
}

void ask(node* root) {
    if (root == NULL) return;

    if (root->left == NULL && root->right == NULL) {
        printf("\n====================================\n");
        printf("%s\n", root->text);
        printf("====================================\n");
        return;
    }

    int choice;
    printf("\n------------------------------------\n");
    printf("%s", root->text);
    printf("\nEnter choice (1 or 2): ");
    scanf("%d", &choice);

    if (choice == 1) ask(root->left);
    else if (choice == 2) ask(root->right);
    else printf("Invalid choice!\n");
}
```

```
int main() {
    node* climate = makeNode("Choose Climate:\n1. Hot/Tropical\n2.
Cool/Temperate");

    node* hot_group = makeNode("Who are you traveling with?\n1.
Solo/Family\n2. Friends/Work");
    node* cool_group = makeNode("Who are you traveling with?\n1.
Solo/Family\n2. Friends/Work");

    climate->left = hot_group;
    climate->right = cool_group;

    node* hot_solo_or_family = makeNode("Choose:\n1. Solo\n2.
Family");
    node* hot_friends_or_work = makeNode("Choose:\n1. Friends\n2.
Work");
    hot_group->left = hot_solo_or_family;
    hot_group->right = hot_friends_or_work;

    node* cool_solo_or_family = makeNode("Choose:\n1. Solo\n2.
Family");
    node* cool_friends_or_work = makeNode("Choose:\n1. Friends\n2.
Work");
    cool_group->left = cool_solo_or_family;
    cool_group->right = cool_friends_or_work;

    // Hot + Solo
    node* hot_solo = makeNode("Do you prefer:\n1.
Adventure/Activities\n2. Relaxation?");
    hot_solo_or_family->left = hot_solo;

    node* hot_solo_adventure = makeNode("Do you want:\n1. Water
Activities\n2. Land Activities?");
    hot_solo->left = hot_solo_adventure;
    hot_solo_adventure->left = makeNode("Recommended: Bali, Thailand
(Diving/Surfing)");
    hot_solo_adventure->right = makeNode("Recommended: Costa Rica,
Peru (Hiking/Trekking)");

    node* hot_solo_relax = makeNode("Do you want:\n1. Beach\n2. Island
Resort?");
    hot_solo->right = hot_solo_relax;
```

```
    hot_solo_relax->left = makeNode("Recommended: Maldives, Goa (Beach
Relaxation)");
    hot_solo_relax->right = makeNode("Recommended: Santorini, Fiji
(Island Resorts)");

    // Hot + Family
    node* hot_family = makeNode("Do you prefer:\n1. Theme
Parks/Entertainment\n2. Nature/Scenic?");
    hot_solo_or_family->right = hot_family;

    node* hot_family_parks = makeNode("Choose:\n1.
Disney/Universal\n2. Other Parks");
    hot_family->left = hot_family_parks;
    hot_family_parks->left = makeNode("Recommended: Orlando Disney,
Tokyo Disneyland");
    hot_family_parks->right = makeNode("Recommended: Singapore
Universal, Hong Kong Ocean Park");

    node* hot_family_nature = makeNode("Do you prefer:\n1. Beach
Destinations\n2. Mountain/Forest?");
    hot_family->right = hot_family_nature;
    hot_family_nature->left = makeNode("Recommended: Kerala, Hawaii
(Beaches)");
    hot_family_nature->right = makeNode("Recommended: Bali Highlands,
Sri Lanka (Mountains)");

    // Hot + Friends
    node* hot_friends = makeNode("Do you prefer:\n1. Party Vibes\n2.
Adventure/Exploration?");
    hot_friends_or_work->left = hot_friends;

    node* hot_friends_party = makeNode("Do you prefer:\n1. Beach
Party\n2. City Nightlife?");
    hot_friends->left = hot_friends_party;
    hot_friends_party->left = makeNode("Recommended: Ibiza, Phuket
(Beach Parties)");
    hot_friends_party->right = makeNode("Recommended: Bangkok, Miami
(City Nightlife)");

    node* hot_friends_adv = makeNode("Do you want:\n1.
Trekking/Outdoor\n2. Water Sports?");
    hot_friends->right = hot_friends_adv;
```

```
    hot_friends_adv->left = makeNode("Recommended: Thailand Trekking,
Costa Rica Rainforest");
    hot_friends_adv->right = makeNode("Recommended: Philippines
Surfing, Bali Diving");

    // Hot + Work
    node* hot_work = makeNode("Do you want:\n1. Urban City
Experience\n2. Quiet Resort for Workation?");
    hot_friends_or_work->right = hot_work;

    node* hot_work_urban = makeNode("Choose:\n1. Modern Business
Cities\n2. Historic Cities?");
    hot_work->left = hot_work_urban;
    hot_work_urban->left = makeNode("Recommended: Singapore, Dubai
(Modern)");
    hot_work_urban->right = makeNode("Recommended: Miami, Istanbul
(Historic)");

    node* hot_work_resort = makeNode("Choose:\n1. Beach Resort\n2.
Island Retreat?");
    hot_work->right = hot_work_resort;
    hot_work_resort->left = makeNode("Recommended: Bali, Mauritius
(Beach Resorts)");
    hot_work_resort->right = makeNode("Recommended: Maldives,
Seychelles (Island Retreat)");

    // Cool + Solo
    node* cool_solo = makeNode("Do you prefer:\n1. Adventure
(Mountains/Waterfalls)\n2. Relaxation (Lakes/Resorts)?");
    cool_solo_or_family->left = cool_solo;

    node* cool_solo_adv = makeNode("Do you want:\n1. Mountains\n2.
Waterfalls?");
    cool_solo->left = cool_solo_adv;
    cool_solo_adv->left = makeNode("Recommended: Swiss Alps, Nepal
Himalayas (Mountains)");
    cool_solo_adv->right = makeNode("Recommended: Iceland, Norway
(Waterfalls)");

    node* cool_solo_relax = makeNode("Do you want:\n1. Lakes/Scenic
Views\n2. Spa Resorts?");
    cool_solo->right = cool_solo_relax;
```

```
    cool_solo_relax->left = makeNode("Recommended: Canada Rockies,
Norway Fjords (Scenic)");
    cool_solo_relax->right = makeNode("Recommended: New Zealand,
Austria (Spa Resorts)");

    // Cool + Family
    node* cool_family = makeNode("Do you prefer:\n1. Theme Parks\n2.
Nature/Scenic?");
    cool_solo_or_family->right = cool_family;

    node* cool_family_parks = makeNode("Choose:\n1. Disneyland\n2.
Other Parks?");
    cool_family->left = cool_family_parks;
    cool_family_parks->left = makeNode("Recommended: Paris Disneyland,
Tokyo Disneyland");
    cool_family_parks->right = makeNode("Recommended: Legoland
Denmark, Europa Park Germany");

    node* cool_family_nature = makeNode("Do you want:\n1.
Mountains\n2. Countryside?");
    cool_family->right = cool_family_nature;
    cool_family_nature->left = makeNode("Recommended: Switzerland
Alps, Austria (Mountains)");
    cool_family_nature->right = makeNode("Recommended: Scotland,
Ireland (Countryside)");

    // Cool + Friends
    node* cool_friends = makeNode("Do you prefer:\n1. Party
(Festivals/Nightlife)\n2. Adventure (Snow Activities)?");
    cool_friends_or_work->left = cool_friends;

    node* cool_friends_party = makeNode("Do you want:\n1. Summer
Festivals\n2. Nightlife?");
    cool_friends->left = cool_friends_party;
    cool_friends_party->left = makeNode("Recommended: Tomorrowland
Belgium, Glastonbury UK");
    cool_friends_party->right = makeNode("Recommended: Berlin,
Amsterdam (Nightlife)");

    node* cool_friends_adv = makeNode("Do you want:\n1. Mountains\n2.
Snow Sports?");
    cool_friends->right = cool_friends_adv;
```

```
    cool_friends_adv->left = makeNode("Recommended: Nepal, Patagonia
(Mountains)");
    cool_friends_adv->right = makeNode("Recommended: Canadian Rockies,
Swiss Alps (Snow Sports)");

    // Cool + Work
    node* cool_work = makeNode("Do you want:\n1. Urban City
Experience\n2. Quiet Resort for Workation?");
    cool_friends_or_work->right = cool_work;

    node* cool_work_urban = makeNode("Choose:\n1. Major Cities\n2.
Smaller Cities?");
    cool_work->left = cool_work_urban;
    cool_work_urban->left = makeNode("Recommended: London, New York,
Tokyo (Major)");
    cool_work_urban->right = makeNode("Recommended: Vienna, Geneva
(Smaller)");

    node* cool_work_resort = makeNode("Choose:\n1. Mountain
Resorts\n2. Countryside Retreat?");
    cool_work->right = cool_work_resort;
    cool_work_resort->left = makeNode("Recommended: Swiss Alps, Banff
Canada (Mountain Resorts)");
    cool_work_resort->right = makeNode("Recommended: Iceland, Finland
(Countryside Retreat)");

    printf("====================================\n");
    printf("Travel Recommendation System\n");
    printf("====================================\n");
    ask(climate);

    return 0;
}
```

Sample Input/Output Screenshots:

Input/Output 1:

```
====================================
Travel Recommendation System
====================================

------------------------------------
Choose Climate:
1. Hot/Tropical
2. Cool/Temperate
Enter choice (1 or 2): 1

------------------------------------
Who are you traveling with?
1. Solo/Family
2. Friends/Work
Enter choice (1 or 2): 1

------------------------------------
Choose:
1. Solo
2. Family
Enter choice (1 or 2): 1
```

```
------------------------------------
Do you prefer:
1. Adventure/Activities
2. Relaxation?
Enter choice (1 or 2): 2

------------------------------------
Do you want:
1. Beach
2. Island Resort?
Enter choice (1 or 2): 2

====================================
Recommended: Santorini, Fiji (Island Resorts)
====================================
```

Input/Output 2:

```
====================================
Travel Recommendation System
====================================

------------------------------------
Choose Climate:
1. Hot/Tropical
2. Cool/Temperate
Enter choice (1 or 2): 1

------------------------------------
Who are you traveling with?
1. Solo/Family
2. Friends/Work
Enter choice (1 or 2): 1

------------------------------------
Choose:
1. Solo
2. Family
Enter choice (1 or 2): 2
```

```
------------------------------------
Do you prefer:
1. Theme Parks/Entertainment
2. Nature/Scenic?
Enter choice (1 or 2): 2

------------------------------------
Do you prefer:
1. Beach Destinations
2. Mountain/Forest?
Enter choice (1 or 2): 1

====================================
Recommended: Kerala, Hawaii (Beaches)
====================================
```

Input/Output 3:

```
====================================
Travel Recommendation System
====================================

----------------------------------
Choose Climate:
1. Hot/Tropical
2. Cool/Temperate
Enter choice (1 or 2): 2

----------------------------------
Who are you traveling with?
1. Solo/Family
2. Friends/Work
Enter choice (1 or 2): 2

----------------------------------
Choose:
1. Friends
2. Work
Enter choice (1 or 2): 1
```

```
----------------------------------
Do you prefer:
1. Party (Festivals/Nightlife)
2. Adventure (Snow Activities)?
Enter choice (1 or 2): 2

----------------------------------
Do you want:
1. Mountains
2. Snow Sports?
Enter choice (1 or 2): 2

====================================
Recommended: Canadian Rockies, Swiss Alps (Snow Sports)
====================================
```

Input/Output 4:

```
====================================
Travel Recommendation System
====================================

------------------------------------
Choose Climate:
1. Hot/Tropical
2. Cool/Temperate
Enter choice (1 or 2): 2

------------------------------------
Who are you traveling with?
1. Solo/Family
2. Friends/Work
Enter choice (1 or 2): 2

------------------------------------
Choose:
1. Friends
2. Work
Enter choice (1 or 2): 2
------------------------------------
Do you want:
1. Urban City Experience
2. Quiet Resort for Workation?
Enter choice (1 or 2): 1

------------------------------------
Choose:
1. Major Cities
2. Smaller Cities?
Enter choice (1 or 2): 2

====================================
Recommended: Vienna, Geneva (Smaller)
====================================
```

Result:

The Travel Recommendation System was successfully implemented using the C programming language with a binary tree structure to represent a decision-making process. Upon execution, the program:

- Displays a welcome message to introduce the system.
- Prompts the user with a series of multiple-choice questions related to:
    - Climate preference (Hot or Cool),
    - Travel group type (Solo, Family, Friends, or Work),
    - Personal interests (Adventure, Relaxation, Entertainment, etc.).
- Traverses a binary tree based on user input, guiding them step-by-step through the decision path.
- Presents a final travel destination recommendation at the leaf node based on the user's choices.

Conclusion:

The Travel Recommendation System project demonstrates the practical application of **binary trees** in a real-world decision-making scenario. The system guides the user through a structured series of questions and uses their responses to navigate through the tree, ultimately arriving at a personalized travel destination suggestion.

The project fulfills its primary objectives:

- It provides an interactive and user-friendly experience.
- It applies fundamental C programming concepts such as structures, dynamic memory allocation, recursion, and input handling.
- It models a simplified version of recommendation systems used in modern travel platforms.

This project serves as a foundational implementation that can be further extended by integrating more features such as:

- File input/output for dynamic tree creation,
- Graphical user interface (GUI),
- Database or API connectivity,
- Support for more than two options per question (multi-way decision trees).

In conclusion, the system is a successful demonstration of how logical decision trees can be applied to create useful, interactive tools that assist users in personalized decision-making.

References:

- C Programming Language — Kernighan & Ritchie
- GeeksforGeeks (Binary Trees, Recursion examples)
- TutorialsPoint (C Pointers and Structures)