|  |
| --- |
| **Data Structures and Algorithms** |
| Connecting Minds, Igniting Hearts <3 |
| **Course Project Report** |

|  |
| --- |
| **School of Computer Science and Engineering**  **2023-24** |

**Contents**

|  |  |
| --- | --- |
| **Si. No.** | **Topics** |
| 1. | Course and Team Details |
| 2. | Introduction |
| 3. | Problem Definition |
| 4. | Functionality Selection |
| 5. | Functionality Analysis |
| 6. | Conclusion |
| 7. | References |

**1. Course and Team Details**

**1.1 Course details**

|  |  |
| --- | --- |
| **Course Name** | Data Structures and Algorithms |
| **Course Code** |  |
| **Semester** | III |
| **Division** | C |
| **Year** | 2023-24 |
| **Instructor** |  |

**1.2 Team Details**

|  |  |  |
| --- | --- | --- |
| **Si. No.** | **Roll No.** | **Name** |
| 1. | 337 | Bhavana Shripad Gombi |
| 2. | 340 | Aditi Javali |
| 3. | 343 | Shreya Patil |
| 4. | 345 | Mallikarjun Honnalli |

**1.3 Report Owner**

|  |  |
| --- | --- |
| **Roll No.** | **Name** |
| 343 | Shreya Patil |

**2. Introduction**

The selected project focuses on the development of a social platform facilitating connections between individuals based on shared interests and compatibility factors. In this domain, users register on the platform, providing information about their interests, preferences, and personal details, allowing the system to suggest compatible matches. The project integrates advanced algorithms, such as Rabin-Karp and Boyer-Moore, for efficient username validation and searching within the user database. Additionally, it incorporates a recommendation system that calculates compatibility scores between users, employing Dijkstra's algorithm for optimal matchmaking.

The inspiration for this project stems from a white paper [1] that delves into the dynamics of social network platforms, emphasizing the significance of compatibility matching algorithms in enhancing user engagement and satisfaction. The paper discusses the challenges faced by traditional social networks in establishing meaningful connections and presents algorithmic solutions for overcoming these challenges. The utilization of advanced string-searching algorithms, as demonstrated in the white paper, influenced the incorporation of Rabin-Karp and Boyer-Moore algorithms in username validation to enhance user experience.

**3. Problem Statement**

**3.1 Domain**

The problem being addressed in this code is related to a social networking and matchmaking platform where users can register, log in, find compatible matches, and perform various operations related to user accounts. The system also includes a feature for searching and displaying information about cafes. The motivation behind this problem is to automate the management of user profiles, facilitate user registration and login, implement a matchmaking algorithm based on user compatibility, and provide information about cafes through a binary search tree (BST).

Overall, the problem statement involves creating a program that simulates a social networking platform with matchmaking features and cafe information retrieval, addressing the needs outlined in the white paper.

**3.2 Module Description**

The code includes functionalities such as user registration, login, finding compatible matches, checking user registration status, deleting user accounts, searching for cafes, and an option to exit the program.

The code leverages data structures like arrays, structures, and a binary search tree for efficient storage and retrieval of user and cafe information. The user profiles are saved and loaded from a file, allowing persistence between program executions. The cafe information is organized in a BST for quick search and retrieval.

**4. Functionality Selection**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Si. No.** | **Functionality Name** | **Known** | **Unknown** | **Principles applicable** | **Algorithms** | **Data Structures** |
|  | Name the functionality within the module | What information do you already know about the module? What kind of data you already have? How much of process information is known? | What are the pain points? What information needs to be explored and understood? What are challenges? | What are the supporting principles and design techniques? | List all the algorithms you will use | What are the supporting data structures? |
| 1 | Find Top Three Matches. | Each user is a node in the graph, and edges between nodes represent compatibility. The compatibility between users is determined based on shared interests and preferences. | The code doesn't reveal the specific details of how user compatibility is calculated. The code doesn't explicitly handle edge cases or error scenarios. | Prioritize the user experience by understanding user needs, preferences, and behaviors. | Utilizes Dijkstra's algorithm to find compatibility scores and presents the top three matches and **quick sort** is used for sorting. | Arrays and Linked list. |
| 2 | Check whether the user is present. | Function takes user input for a name to be searched. | Different strings producing the same hash value. |  | **Rabin-Karp stringmatching.** | Arrays. |
| 3 | Delete the user. | Deletes a user account by shifting the array and reducing the user count. | Details of how the user account is removed from memory. |  | **Brute Force Stringmatching.** | Arrays. |
| 4 | Search Cafe | Handles operations on a Binary Search Tree for cafe information | Specifics of how the BST is maintained and utilized for cafe information. | Unique cafes | **Binary Search Tree nodes.** | Linked list and trees. |
| … |  |  |  |  |  |  |
| … |  |  |  |  |  |  |

**5. Functionality Analysis**

### 1. User Registration Module:

**Workflow:**

* The module prompts the user to enter various details and then registers the user.
* Utilizes the Boyer-Moore string searching algorithm to ensure the uniqueness of the username.

**Efficiency Analysis:**

* Boyer-Moore algorithm has an average and worst-case time complexity of O(n + m), where n is the length of the text, and m is the length of the pattern.
* In the worst-case scenario, the algorithm would iterate through the entire list of registered usernames.
* The overall efficiency is acceptable for typical sizes of registered user lists.

### 2. User Login Module:

**Workflow:**

* Prompts the user for their username and password.
* Performs a linear search to find a matching username and password combination.

**Efficiency Analysis:**

* Linear search has a time complexity of O(n), where n is the number of registered users.
* This is reasonable for small to medium-sized user databases.

### 3. Compatibility Calculation Module:

**Workflow:**

* Computes a compatibility score between two user profiles.
* Utilizes various factors such as gender, interests, and preferences.

**Efficiency Analysis:**

* The algorithm involves comparisons and assignments with constant time complexity for each factor.
* The overall efficiency is linear with a small constant factor.

### 4. User Deletion Module:

**Workflow:**

* Deletes a user account by specifying the username.
* Performs a simple linear search **Brute-force algorithm** to find the user to delete.

**Efficiency Analysis:**

* Linear search has a time complexity of O(n), where n is the number of registered users.
* This is reasonable for small to medium-sized user databases.

### 5. Cafe Information Retrieval Module:

**Workflow:**

* Reads cafe information from a file into a Binary Search Tree (BST).
* Searches for a cafe in the BST based on the cafe name.

**Efficiency Analysis:**

* The insertion and search operations in a balanced BST have an average time complexity of O(log n), where n is the number of cafes.
* This is efficient for managing cafe information.

### 6. Save and Load User Profiles Module:

**Workflow:**

* Writes user profiles to a file and reads user profiles from a file.

**Efficiency Analysis:**

* The writing and reading operations have a linear time complexity relative to the number of users.
* The efficiency is reasonable for moderate-sized user databases.

### 7. Main Menu Module:

**Workflow:**

* Presents a menu to the user, and based on the user's choice, calls the corresponding module.

**Efficiency Analysis:**

* The menu system has constant time complexity for each selection.
* The efficiency is not a concern, as the number of menu options is typically small.

**6. Conclusion**

1. **Understanding of Algorithms and Data Structures:**
   * The project involves implementing and utilizing algorithms like Boyer-Moore string searching, sorting (Quick Sort), and data structures like Binary Search Trees (BST).
   * Learners can gain a deeper understanding of how these algorithms and data structures work and when to apply them.
2. **User Input Handling and Validation:**
   * Handling user input is a crucial aspect of any application. This project involves taking user input for various fields, validating it, and providing appropriate feedback.
   * Learners can understand the importance of input validation for maintaining data integrity and preventing errors.
3. **File Handling Operations:**
   * Loading and saving user profiles from/to a file is a common practice in software development. This project provides hands-on experience with file operations.
   * Learners can understand how to read from and write to files, which is essential for persistent data storage.
4. **Menu-Driven User Interface:**
   * Implementing a menu-driven user interface is a practical way to interact with users. This project demonstrates how to structure and manage user interactions through a menu system.
   * Learners can gain insights into creating user-friendly interfaces and handling user choices.
5. **Project Organization and Modularity:**
   * The project is organized into distinct modules, each responsible for specific functionalities (e.g., user registration, login, cafe information retrieval).
   * Learners can understand the importance of modular programming, making the code more readable, maintainable, and easy to debug.
6. **Efficiency Considerations:**
   * The project involves analyzing the efficiency of algorithms and data structures chosen for specific tasks, such as searching and sorting.
   * Learners can develop a sense of when to prioritize efficiency and choose appropriate algorithms based on the requirements.
7. **Real-world Application:**
   * The project simulates a simplified social matching system and cafe information retrieval system, reflecting real-world scenarios where multiple functionalities need to work together.
   * Learners can see how individual components come together to form a cohesive application.
8. **Error Handling and User Feedback:**
   * The project includes error handling mechanisms and provides feedback to users in case of invalid input or unsuccessful operations.
   * Learners can understand the importance of graceful error handling for a better user experience.

**7. References**

[1] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. 2001. Introduction to Algorithms (2nd ed.). McGraw-Hill Higher Education.

**~\*~\*~\*~\*~\*~\*~\*~**