DEPARTMENT OF COMPUTER & INFORMATION SYSTEMS ENGINEERING
BACHELORS IN COMPUTER SYSTEMS ENGINEERING
Course Code: CS-115
Course Title: Computer Programming

Complex Engineering Problem

FE Batch 2024, Fall Semester 2024
Grading Rubric
TERM PROJECT

Group Members:

Student No.	Name	Roll No.
S1	Ukasha	CS-24115
S2	Osama	CS-24120
S3	Taha	CS-24122

CRITERIA AND SCALES				**Marks Obtained**		
				S1	**S2**	**S3**
Criterion 1: Does the application meet the desired specifications and produce the desired outputs? (CPA-1, CPA-3) **[8 marks]**						
1	2	3	4			
The application does not meet the desired specifications and is producing incorrect outputs.	The application partially meets the desired specifications and is producing incorrect or partially correct outputs.	The application meets the desired specifications but is producing incorrect or partially correct outputs.	The application meets all the desired specifications and is producing correct outputs.			
Criterion 2: How well is the code organization? **[2 marks]**						
1	2	3	4			
The code is poorly organized and very difficult to read.	The code is readable only to someone who knows what it is supposed to be doing.	Some part of the code is well organized, while some part is difficult to follow.	The code is well organized and very easy to follow.			
Criterion 3: How friendly is the application interface? (CPA-1, CPA-3) **[2 marks]**						
1	2	3	4			
The application interface is difficult to understand and use.	The application interface is easy to understand and but not that comfortable to use.	The application interface is very easy to understand and use.	The application interface is very interesting/ innovative and easy to understand and use.			
Criterion 4: How does the student performed individually and as a team member? (CPA-2, CPA-3) **[4 marks]**						
1	2	3	4			
The student did not work on the assigned task.	The student worked on the assigned task, and accomplished goals partially.	The student worked on the assigned task, and accomplished goals satisfactorily.	The student worked on the assigned task, and accomplished goals beyond expectations.			
Criterion 5: Does the report adhere to the given format and requirements? **[4 marks]**						
1	2	3	4			
The report does not contain the required information and is formatted poorly.	The report contains the required information only partially but is formatted well.	The report contains all the required information but is formatted poorly.	The report contains all the required information and completely adheres to the given format.			
			Total Marks:			

Teacher's Signature

DATABASE MANGMENT SYSTEM

Problem Description

The aim of this project is to develop a simplified Database Management System (DBMS) using Python. The DBMS allows users to create, open, and manipulate custom databases with variable fields and field lengths. The system provides functionalities for adding, editing, deleting, viewing, and searching records. The project focuses on creating a user-friendly command-line interface for easy interaction with the database system.

Distinguishing Features of Project

What sets this project apart from other simple DBMS projects is its lightweight structure and scalability for future expansion. While it currently functions as a basic file-based DBMS using Python, its architecture is flexible enough to accommodate future advancements. The project is built on the following core features:

- ***Field-based Database Creation:*** Users can define the structure of their database by specifying field names and their respective lengths.
- ***Persistent Storage:*** Data is stored in JSON format, enabling portability and scalability.
- ***Data Manipulation:*** Users can perform common database operations, such as adding, editing, deleting, viewing, and searching records.
- ***File-based Structure:*** Each database and its metadata (field names and lengths) are saved in separate files, making the system simple to implement while being efficient for small to medium-sized datasets.
- ***Search Records on Any Attribute:*** Users can search for records based on any field name/attribute in the database. This allows for greater flexibility in querying the data. The user selects a field and enters the value they wish to search for, and the system will retrieve and display all matching records, making it easy to locate specific entries.

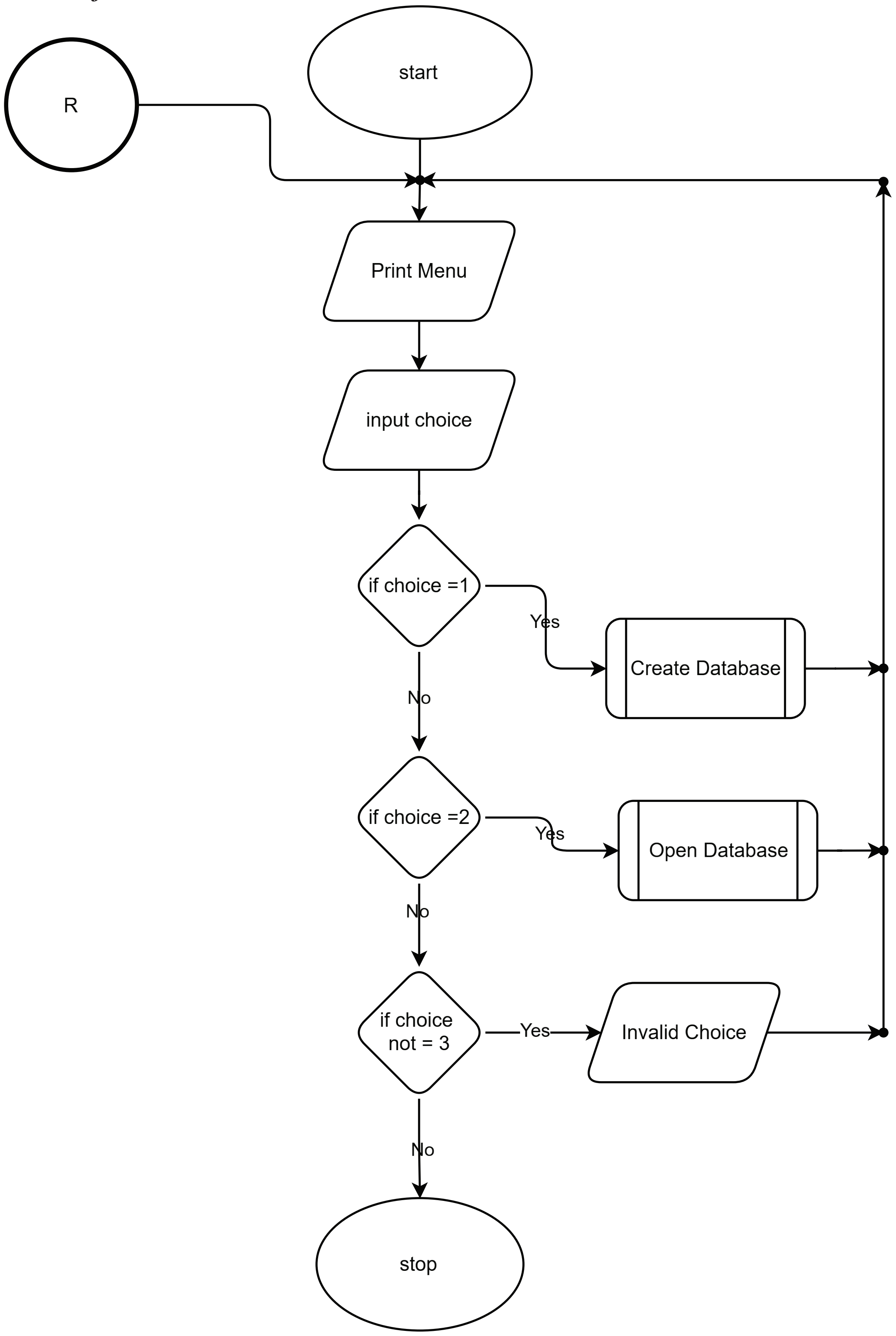
Flow of Your Project

The flow of the project is as follows:

- ***Database Creation:*** The user is prompted to enter a name for the new database and field names with their corresponding lengths. The metadata is saved in a JSON file, and the database name is added to a central list of databases stored in DATABASES.txt.
- ***Database Opening:*** The user can open an existing database. The system loads metadata and existing records from JSON files. The user can then perform various operations like adding, editing, deleting, viewing, and searching records.
- ***Record Manipulation:*** Each operation is done by interacting with the command-line interface, guiding the user through adding/editing/deleting/viewing/searching records.
- ***Exit:*** The program provides an option to exit, saving all changes to the database.

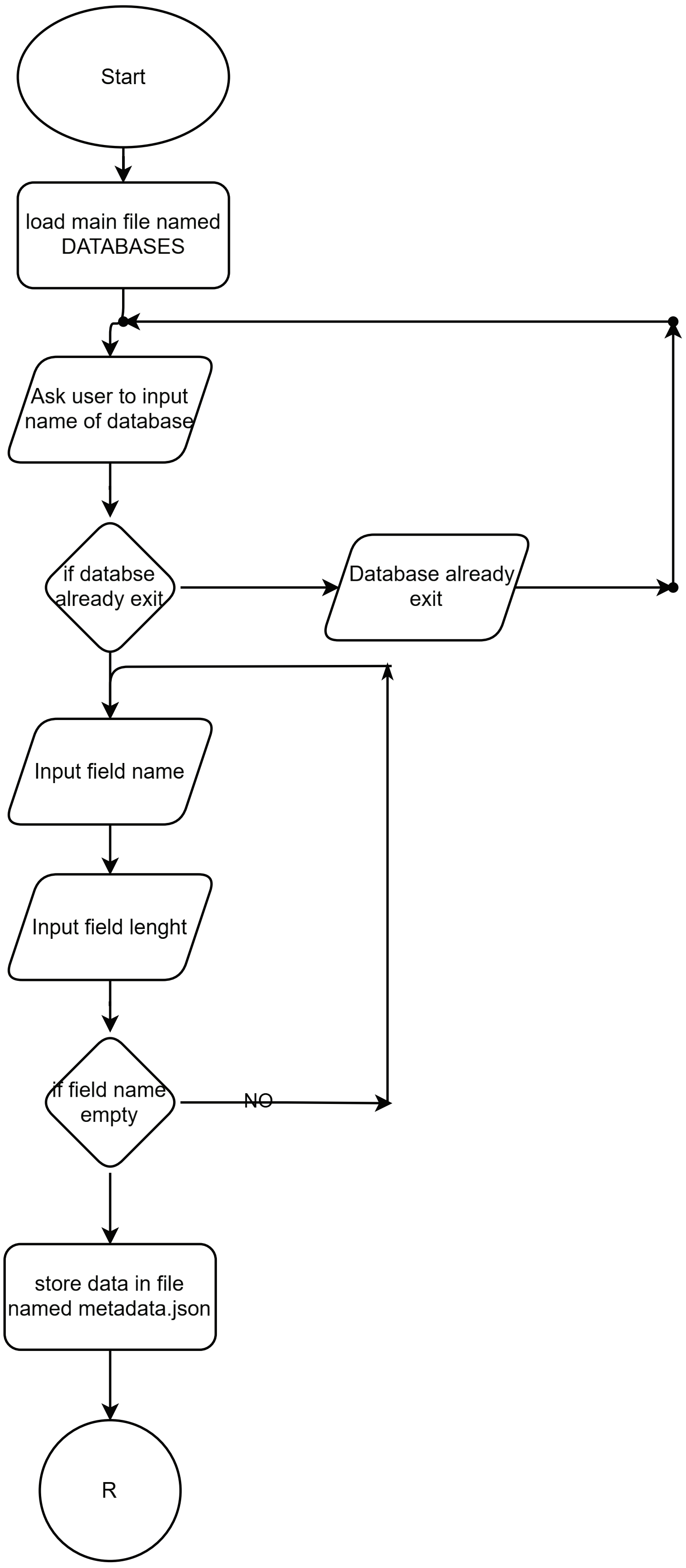
DATABASE MANGMENT SYSTEM

■ *Flow of Main Module*



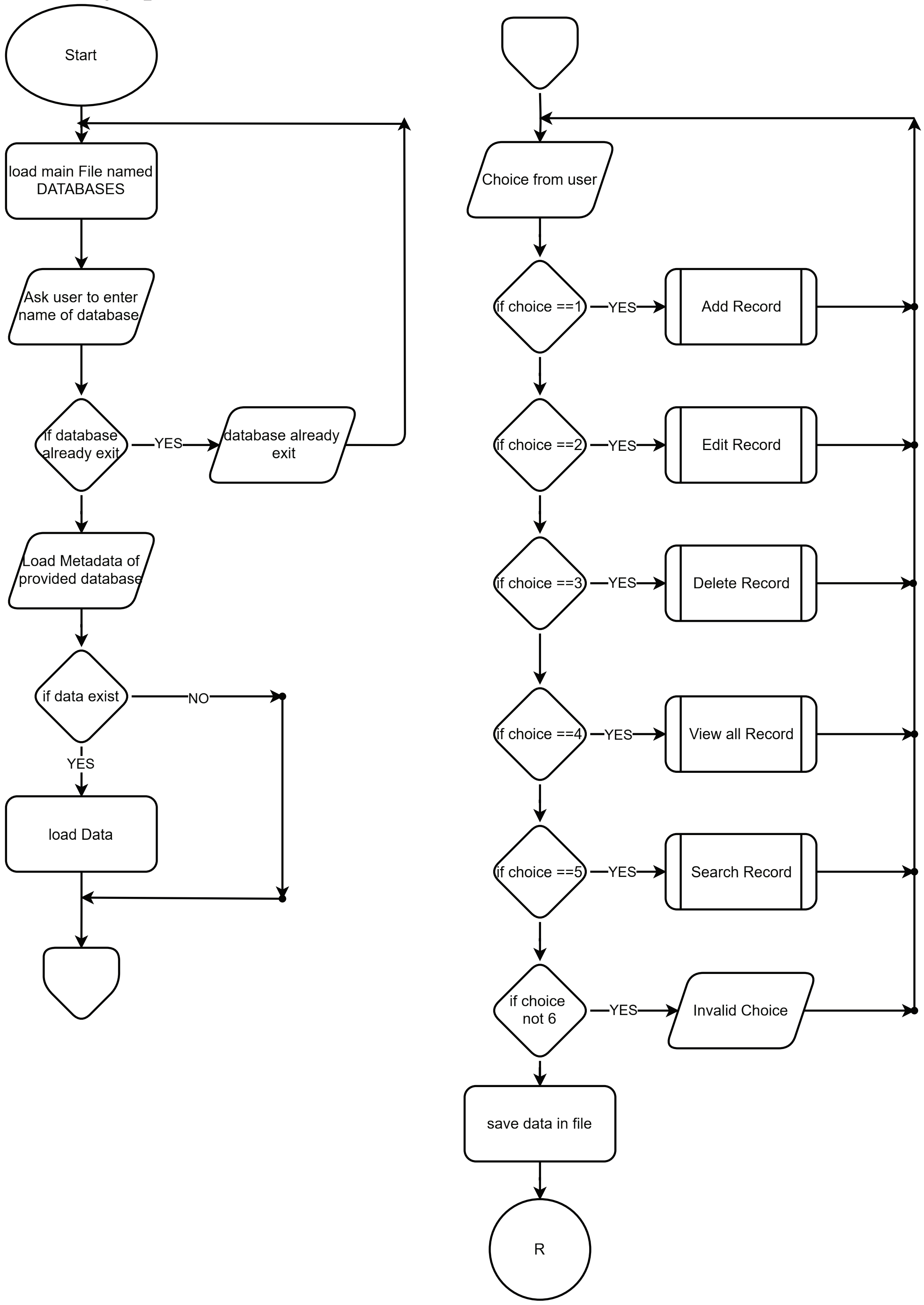
DATABASE MANGMENT SYSTEM

Flow of Create Module



DATABASE MANAGEMENT SYSTEM

Flow of Open Module



■ *Most Challenging Part*

The most challenging aspect of the project was implementing the Search Record functionality. While basic search seems straightforward, our goal was to create a flexible system that allows users to query records based on any attribute. This involved handling dynamic field selection, user input, and ensuring accurate results . The interface needed to let users easily choose a field and input a search value. The most satisfying outcome was enabling dynamic attribute selection, allowing users to search within any field. This feature enhanced the DBMS's flexibility, providing MongoDB-like querying capabilities in a lightweight, file-based system.

■ *New Things Learned*

Working on this project introduced several new concepts and techniques that were applied in practical ways

- ◆***File Handling with JSON:*** We learned how to store and retrieve data in JSON format, which is an efficient way to persist data in a human-readable structure. This was crucial for storing both metadata (field names and lengths) and actual data records.

- ◆***Data Validation:*** We applied multiple techniques to validate user input, ensuring that data entered by users did not violate any constraints (e.g., field lengths). This helped us maintain data integrity.

- ◆***Recursion:*** In certain functions, we used recursion (e.g., when a user attempts to create a database that already exists), which allowed us to handle repeated operations efficiently.

- ◆***Try and except:*** we also learned about error handling in Python using try and except blocks. This concept became particularly useful when working with file handling and ensuring that our program could handle errors gracefully.

■ *Future Expansions*

- ◆***GUI Integration:*** A graphical user interface could be developed for more user-friendly interaction with the DBMS.

- ◆***Security Features:*** Implementing user authentication and access control to manage who can modify or view the database.

- ◆***Primary Field for Data Integrity:*** The concept of a "primary field" (or primary key) for each table or database can be introduced. This field would uniquely identify each record and prevent the insertion of duplicate records.

Contributions of Each Group Member

◆ ***UKASHA (CS-24115):***

1. Core Development: Oversaw the overall program flow and structure.
2. Error Handling and Debugging: Identified and resolved errors in the code.
3. File Handling and Search: Implemented file operations and the search record functionality.

◆ ***OSAMA (CS-24120):***

1. Database Creation and Record Management: Developed functions for creating databases and adding records.
2. Record Viewing: Implemented the functionality to display stored records.

◆ ***TAHA (CS-24122):***

1. Record Editing and Deletion: Created functions to modify and remove existing records.
2. Search and Open: Implemented the search functionality and the process of opening existing databases.

List of References

◆ ***JSON Module Documentation:*** https://docs.python.org/3/library/json.html

◆ ***Working With JSON Data in Python:*** : https://realpython.com/python-json/

◆ ***All About DBMS:*** https://www.geeksforgeeks.org/getting-started-with-database-management-system/

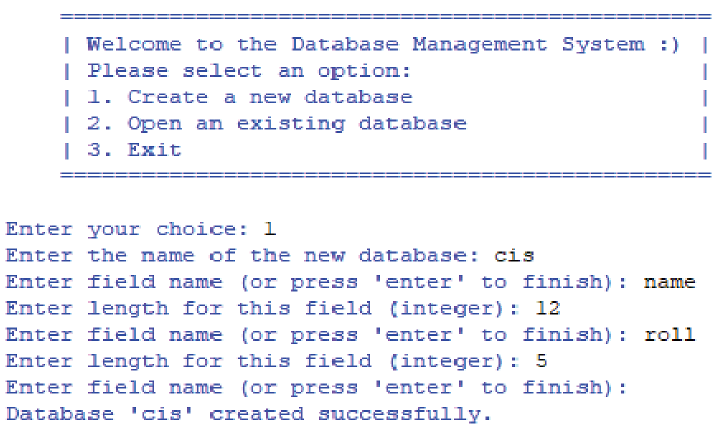
Test Case Runs

Test Case 1: Create Database and Add Records

Action: Create a new database with fields.

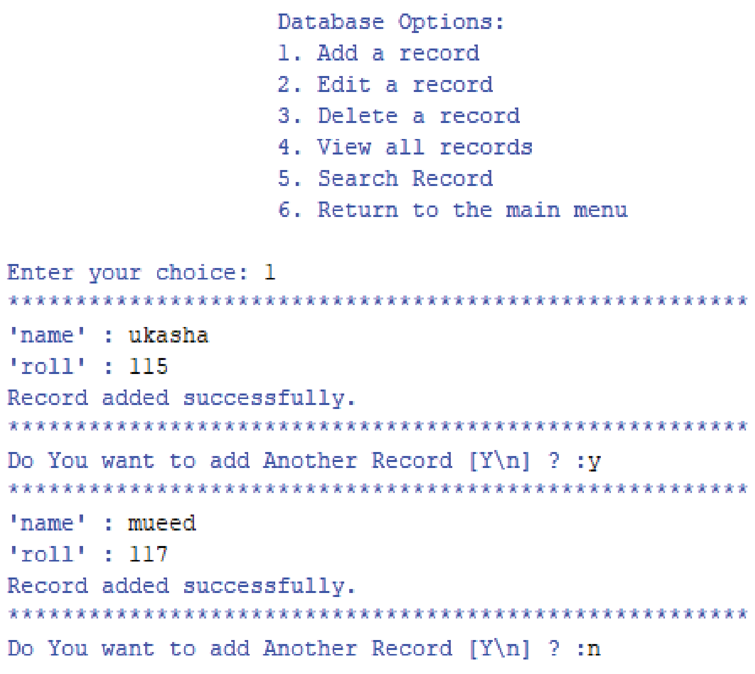
Expected Outcome: Database is created successfully, and records are added.

Screenshot:



```
================================================
| Welcome to the Database Management System :)  |
| Please select an option:                      |
| 1. Create a new database                      |
| 2. Open an existing database                  |
| 3. Exit                                       |
================================================

Enter your choice: 1
Enter the name of the new database: cis
Enter field name (or press 'enter' to finish): name
Enter length for this field (integer): 12
Enter field name (or press 'enter' to finish): roll
Enter length for this field (integer): 5
Enter field name (or press 'enter' to finish):
Database 'cis' created successfully.
```



```
                    Database Options:
                    1. Add a record
                    2. Edit a record
                    3. Delete a record
                    4. View all records
                    5. Search Record
                    6. Return to the main menu

Enter your choice: 1
*******************************************************
'name' : ukasha
'roll' : 115
Record added successfully.
*******************************************************
Do You want to add Another Record [Y\n] ? :y
*******************************************************
'name' : mueed
'roll' : 117
Record added successfully.
*******************************************************
Do You want to add Another Record [Y\n] ? :n
```

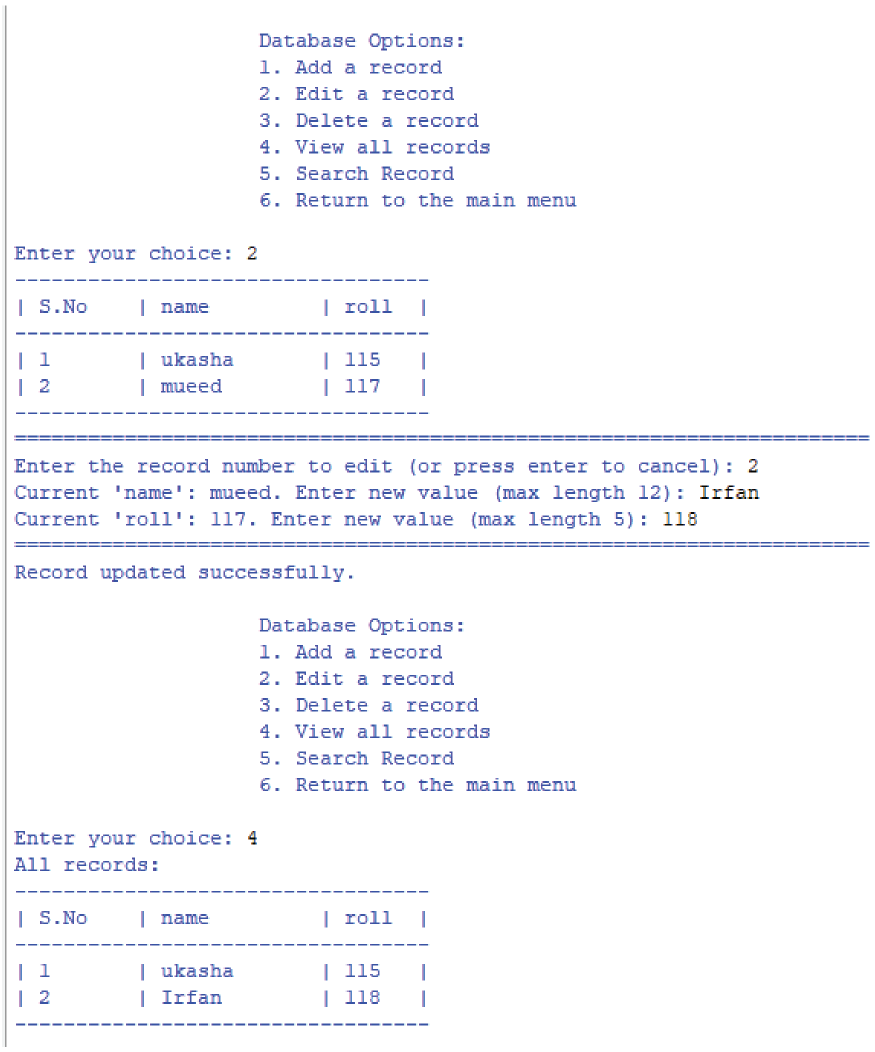
Test Case Runs

◆*Test Case 2: View Records and Edit Records*

Action: Edit a previously added record.

Expected Outcome: The record is updated with the new value

Screenshot:



```
                    Database Options:
                    1. Add a record
                    2. Edit a record
                    3. Delete a record
                    4. View all records
                    5. Search Record
                    6. Return to the main menu

Enter your choice: 2
----------------------------------
| S.No    | name          | roll  |
----------------------------------
| 1       | ukasha        | 115   |
| 2       | mueed         | 117   |
----------------------------------
======================================================================
Enter the record number to edit (or press enter to cancel): 2
Current 'name': mueed. Enter new value (max length 12): Irfan
Current 'roll': 117. Enter new value (max length 5): 118
======================================================================
Record updated successfully.

                    Database Options:
                    1. Add a record
                    2. Edit a record
                    3. Delete a record
                    4. View all records
                    5. Search Record
                    6. Return to the main menu

Enter your choice: 4
All records:
----------------------------------
| S.No    | name          | roll  |
----------------------------------
| 1       | ukasha        | 115   |
| 2       | Irfan         | 118   |
----------------------------------
```

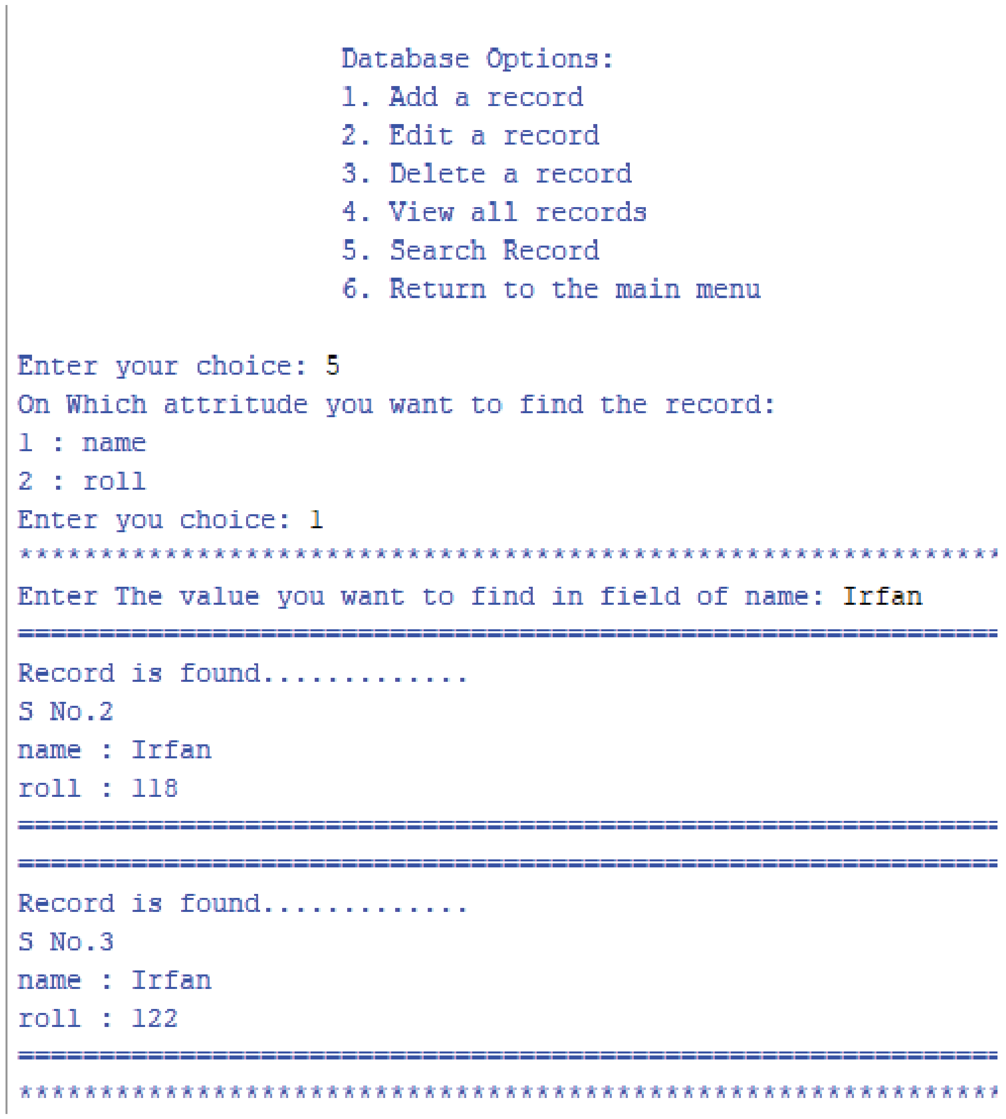
Test Case Runs

◆Test Case 3: Search Record

Action: Search for a record where .

Expected Outcome: The system should display the matching record(s).

Screenshot:



```
                    Database Options:
                    1. Add a record
                    2. Edit a record
                    3. Delete a record
                    4. View all records
                    5. Search Record
                    6. Return to the main menu

Enter your choice: 5
On Which attritude you want to find the record:
1 : name
2 : roll
Enter you choice: 1
**************************************************************
Enter The value you want to find in field of name: Irfan
==============================================================
Record is found............
S No.2
name : Irfan
roll : 118
==============================================================
==============================================================
Record is found............
S No.3
name : Irfan
roll : 122
==============================================================
**************************************************************
```