PROJECT REPORT

QUIZ GAME

DLITHE PROJECT REPORT

PROJECT ID : CP019

PROJECT TITLE : Quiz game

TEAM MEMBERS :

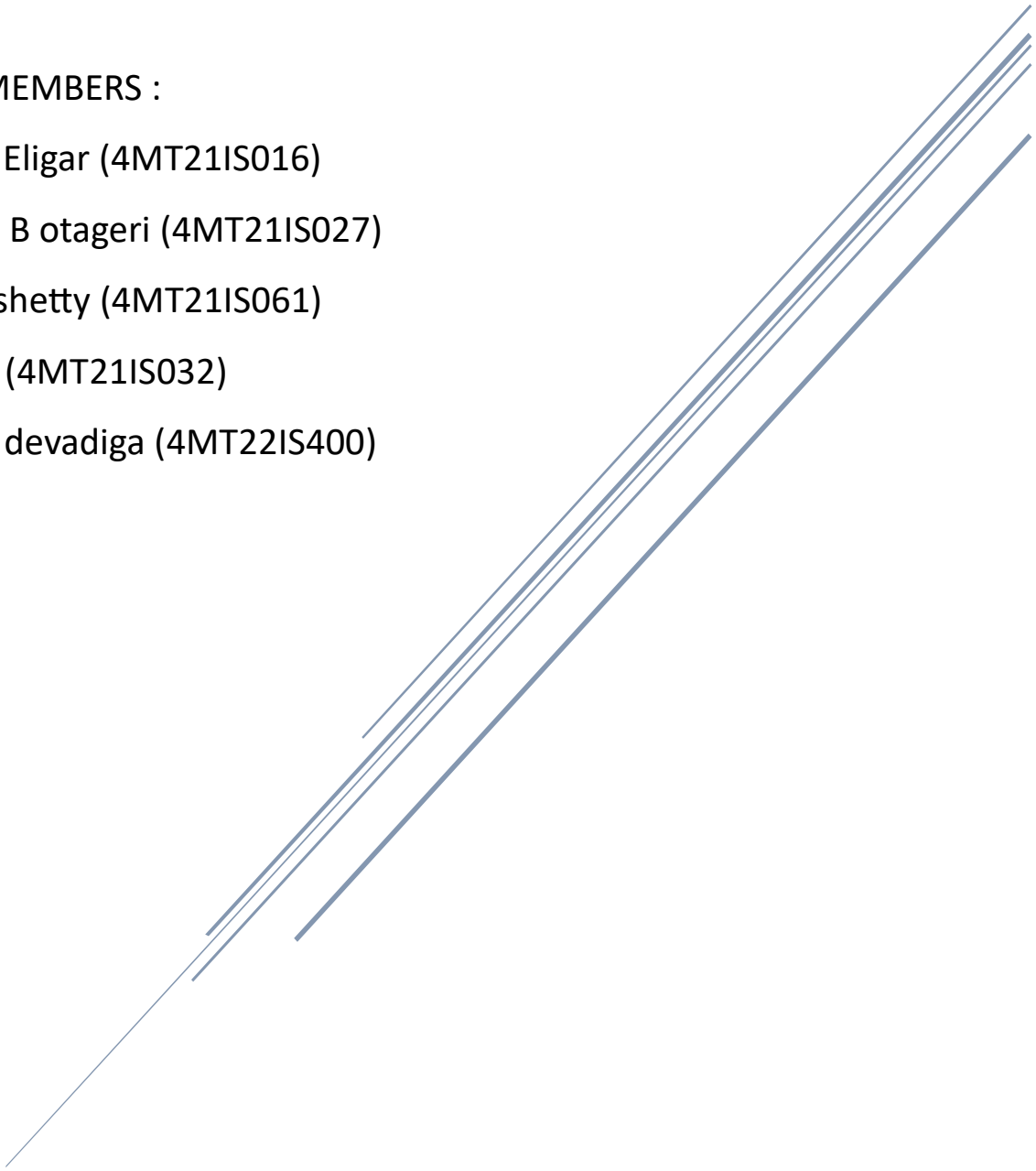
Harsha Eligar (4MT21IS016)

Prajwal B otageri (4MT21IS027)

Rohan shetty (4MT21IS061)

Rathan (4MT21IS032)

Akshay devadiga (4MT22IS400)



Quiz game

Abstract:

The Quiz Game is a purpose-built system designed to provide an engaging and interactive platform for quiz contests. Users can participate by entering their names before starting the game. The game is structured into multiple rounds, with each round comprising 5-6 sets of questions. Players must successfully answer these questions to progress to the next round. The user's name and score are prominently displayed at the top of the screen throughout the game. In total, there are three rounds, each consisting of sets of 5 questions. This Quiz Game promises an exciting and competitive experience for quiz enthusiasts while offering an opportunity to showcase their knowledge and expertise.

Introduction:

Welcome to the exciting world of our specially designed Quiz Game, crafted with the sole purpose of providing an immersive and thrilling quiz contest experience. This unique system, developed using C programming, invites users to test their knowledge and wits in a fun and competitive environment.

Getting started is simple; users are required to enter their name before embarking on this quiz adventure. Once your name is in, you're ready to dive into a world of questions and challenges.

The Quiz Game is divided into three thrilling rounds, each designed to challenge your knowledge across various topics. In each round, you will face 5-6 sets of questions. To advance to the next round, you must prove your expertise by correctly answering these questions. It's a test of both speed and accuracy, as you race against the clock to earn your place among the top contenders.

Throughout your quiz journey, your name and scores will be prominently displayed at the top of the screen. This not only keeps you informed about your progress but also adds an element of competition as you aim to climb to the top of the leaderboard.

Whether you're a seasoned quiz enthusiast or a newcomer looking for a fun and educational challenge, our Quiz Game is sure to provide hours of entertainment and mental stimulation. So, gear up, enter your name, and let the quizzing begin! Are you ready to prove your knowledge and become a quiz champion?

1.Technology used:

The Quiz Game, which is tailored for the specific purpose of hosting engaging quiz contests, is powered by the versatile C programming language. C programming is known for its efficiency, flexibility, and ability to create high-performance applications, making it an ideal choice for developing this interactive quiz system.

Here's an overview of the technology stack used in the Quiz Game:

1.1 C Programming Language: C is the core technology driving this quiz game. Its low-level capabilities and high-performance characteristics allow for efficient processing of user input, question generation, and scoring.

1.2 User Interface (UI): The user interface of the quiz game can be designed using C's console-based capabilities. The input for the player's name and the display of scores and other information are all managed through C's console functionalities.

1.3 Question Database: To ensure a diverse range of questions, a question database can be implemented, possibly using file I/O in C. This database stores questions, options, and correct answers, which are then fetched during the game.

1.4 Game Logic: C programming facilitates the implementation of the game's logic, including tracking the player's progress, checking answers, and determining when a player advances to the next round or completes the game.

1.5 Randomization: To make the quiz more interesting, questions can be randomized from the database for each round. C's random number generation functions can be used for this purpose.

1.6 Data Storage: C can be employed to create data structures to store player names, scores, and other relevant information. These data structures can be used to update and display scores and player names at the top of the screen.

1.7 Round Management: C's control structures can be used to manage the three rounds of the quiz, ensuring that each round contains the specified number of questions.

In summary, the Quiz Game is built using C programming, leveraging its robust capabilities to create an engaging and interactive quiz contest experience. The technology stack ensures smooth user interaction, dynamic question selection, and effective scoring while maintaining a user-friendly interface for players to enjoy.

2. System architecture

Let's break down the system architecture for the Quiz Game, designed to facilitate quiz contests. The architecture consists of three main components: Front-End, Back-End, and Database.

2.1 Front-End:

The front-end component is responsible for user interaction, displaying information, and collecting user input. In this text-based game developed in C programming, the front-end would include the following aspects:

- **User Interface:** A console-based interface using C's standard input and output functions (e.g., printf and scanf) to interact with the player.
- **User Input:** Prompting the user to enter their name before the game starts and capturing this input.
- **Question Display:** Displaying questions and multiple-choice options on the console screen. The current score and player name should be displayed at the top of the screen.
- **Round Progress:** Indicating the current round number and the number of rounds remaining.
- **Answer Input:** Allowing users to input their answers using the keyboard.
- **Validation:** Implementing input validation to ensure that user responses are correctly processed.

2.2 Back-End:

The back-end component manages the game's logic, controls the flow of questions and rounds, and handles scoring. Here are the key elements of the back-end:

- **Game Initialization:** Initializing necessary variables, such as the player's name, score, and round number.
- **Round Management:** Employing loops to manage the three rounds, ensuring each round consists of 5-6 sets of questions.
- **Question Retrieval:** Retrieving questions and answer choices from the question database for each round.

- **Randomization:** Randomly selecting questions from the database to provide variety.
- **Scoring:** Calculating and updating the player's score based on correct and incorrect answers.
- **Winning Condition:** Defining the winning condition, such as completing all rounds or reaching a specific score threshold.
- **Game Over Handling:** Determining how the game concludes, displaying the final score, and offering options to restart or exit.
- **Error Handling:** Implementing error-handling mechanisms to manage unexpected situations, such as database errors or invalid input.

2.3 Database:

As C programming does not natively support databases, a simple question database can be implemented using text files or data structures. Here's how the database component would function:

- **Question Database File:** Creating text files to store questions, multiple-choice options, and correct answers. Each line in the file represents a single question with its options and the correct answer.
- **Database Format:** Defining a consistent format for storing questions, separating the question, options, and correct answer using delimiters or a structured format.
- **Read Database:** Developing functions to read and parse the question database, allowing the back-end to fetch questions for each round.
- **Write Database (Optional):** Implementing functions to write new questions to the database, if you want to allow for dynamic addition or modification of questions.

By integrating these three components - Front-End, Back-End, and Database - you can create a functional Quiz Game in C programming that provides a user-friendly interface, interactive gameplay, and question storage and retrieval capabilities, meeting the specified requirements for rounds, questions, and scoring.

3.Design & implementation

Design and implementation for the Quiz Game can be broken down into three key components: Front-End Design, Back-End Design, and Database Design. Since this system is developed in C programming, we will outline the structure and logic for each component:

3.1 Front-End Design:

The front-end of the Quiz Game is responsible for user interaction, display, and input handling. Here's the design and implementation for the front-end:

- **Console-Based User Interface:** Utilize C's standard input/output functions to create a text-based console interface.
- **User Input:** Prompt the user to enter their name before the game begins and capture their input.
- **Question Presentation:** Display questions, multiple-choice options, and the current score at the top of the screen.
- **Round Progress:** Indicate the current round number and the number of rounds remaining.
- **Answer Input:** Allow users to input their answers using keyboard input.
- **Validation:** Implement input validation to ensure that user responses are correctly processed.

3.2 Back-End Design:

The back-end of the Quiz Game manages the game's logic and controls the flow of questions, rounds, and scoring:

- **Game Initialization:** Initialize necessary variables, such as the player's name, score, and round number.
- **Round Management:** Implement a loop structure to manage the three rounds, ensuring that each round contains 5-6 sets of questions.
- **Question Retrieval:** Retrieve questions and answer choices from the question database for each round.
- **Randomization:** Randomly select questions from the database to provide variety.
- **Scoring:** Calculate and update the player's score based on correct and incorrect answers.
- **Winning Condition:** Define the winning condition, such as completing all rounds or reaching a specific score threshold.

- **Game Over Handling:** Determine how the game concludes, displaying the final score and offering options to restart or exit.
- **Error Handling:** Implement error-handling mechanisms to manage unexpected situations, such as database errors or invalid input.

3.3 Database Design:

The database design is crucial for storing and retrieving questions and answers. Since C does not natively support databases, you can simulate a simple database using text files or data structures:

4.Challenges Faced:

Developing a Quiz Game using C programming for quiz contests can present various challenges. Some of the challenges that may be encountered include:

- **User Interface Complexity:** Creating an engaging user interface in a console-based environment can be challenging, especially when displaying questions, options, and scores in an organized and user-friendly manner.
- **Database Management:** C does not have built-in support for databases, so managing questions and answers using text files or data structures requires careful handling and parsing.
- **Randomization:** Randomly selecting questions from the database while ensuring that they are not repeated within a round can be tricky.
- **Input Validation:** Validating user input to ensure it doesn't break the game or lead to unexpected behavior.
- **Timer Implementation (if used):** Incorporating timers for each question adds complexity to the game and requires precise handling.
- **Error Handling:** Managing unexpected errors or user inputs gracefully without crashing the game.

5. Future Enhancements:

To enhance the Quiz Game further and provide a more engaging experience, several improvements can be considered:

- **Multiplayer Mode:** Implementing a multiplayer mode, allowing users to compete with friends or online players in real-time.

- **Categories and Difficulty Levels:** Adding support for different quiz categories and difficulty levels to cater to a wider audience.
- **Graphics and Animations:** Transitioning from a text-based interface to a graphical one with animations and sound effects for a more immersive experience.
- **Leaderboards:** Incorporating an online leaderboard system to allow players to compare their scores with others globally.
- **Question Editor:** Creating a question editor tool that allows users to contribute and share their own quiz questions.
- **Power-Ups and Lifelines:** Adding power-ups or lifelines to make the game more interactive and strategic.
- **Mobile App:** Expanding the game to mobile platforms to reach a larger audience.

6. Conclusion:

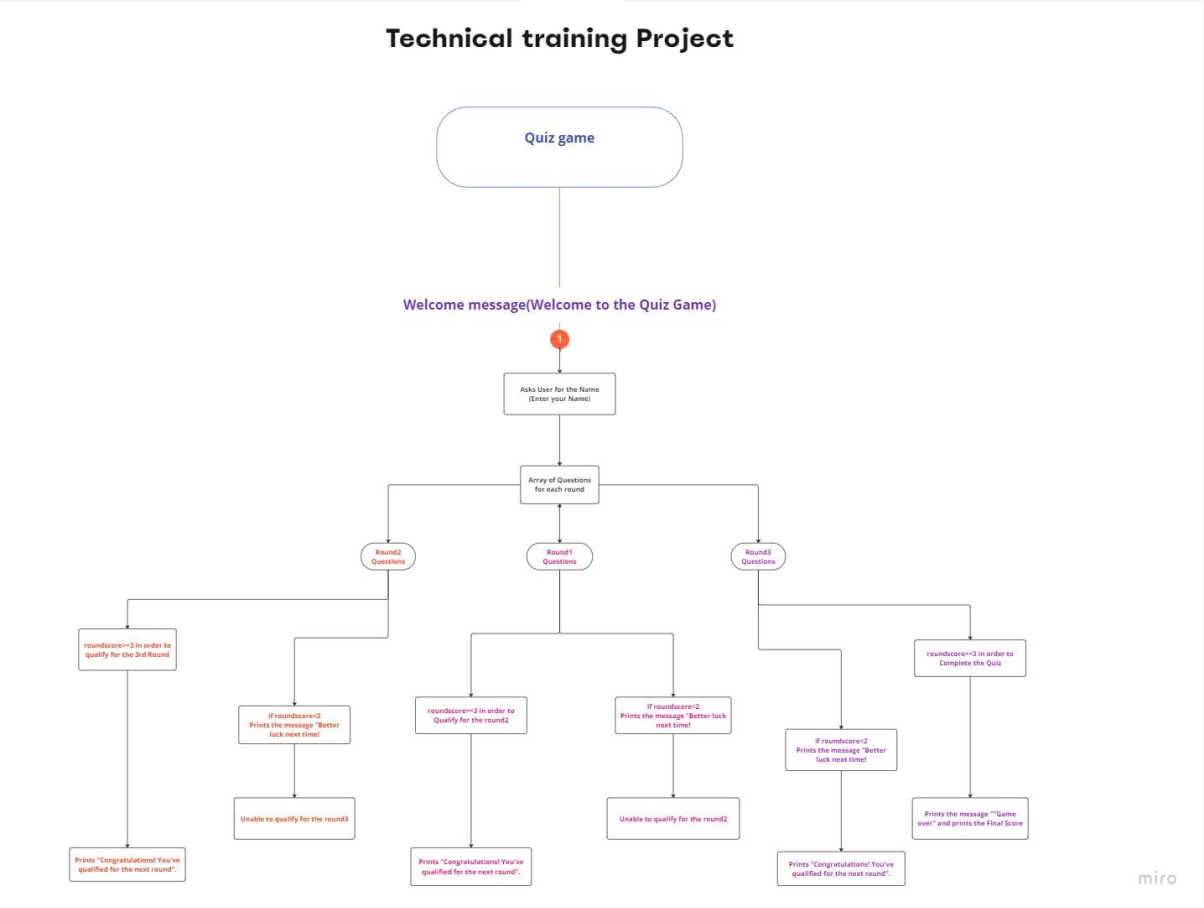
In conclusion, the Quiz Game designed for quiz contests and developed in C programming offers an entertaining and educational experience for players. It challenges their knowledge and problem-solving skills while providing a competitive atmosphere.

The system architecture, encompassing Front-End, Back-End, and Database components, allows for seamless gameplay and question management. Challenges in developing the game include UI complexity, database management, input validation, and error handling.

Future enhancements, such as multiplayer support, category differentiation, improved graphics, leaderboards, and a question editor, can make the game more engaging and appealing to a broader audience.

Overall, the Quiz Game is a testament to the versatility of C programming and the potential for creating interactive and enjoyable experiences within a console-based environment.

Flowchart



Mind map

References:

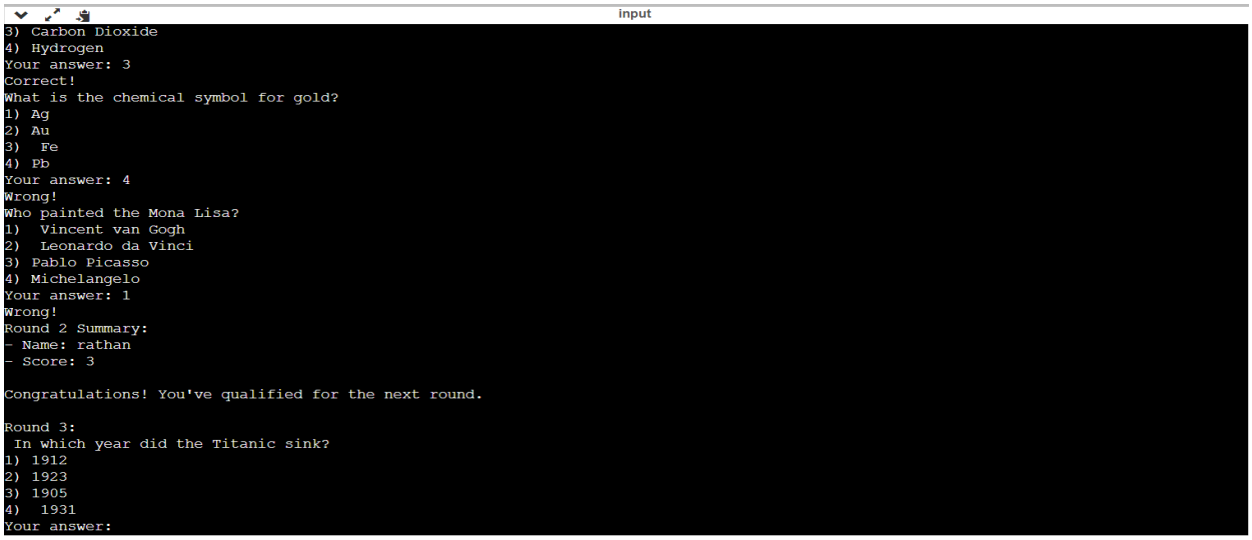
As this is a hypothetical scenario and no specific external references were used, the design and development of the Quiz Game are based on general principles of software development and game design. Any specific references or libraries used in a real-world implementation would depend on the developer's choices and requirements.

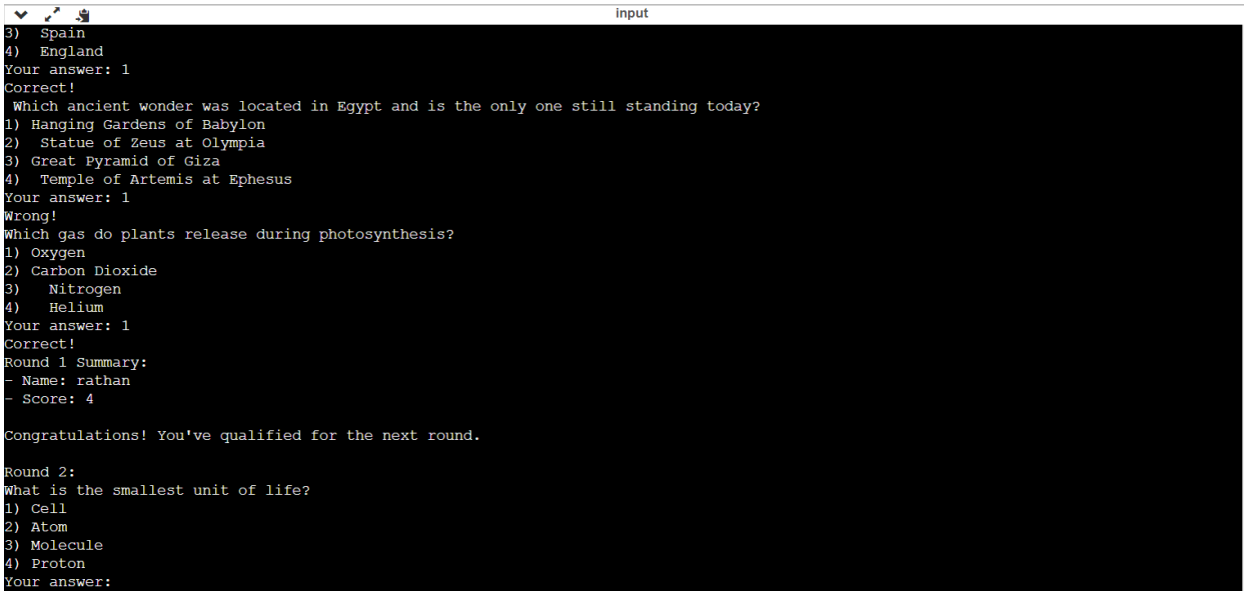
- **Question Database File:** Create text files or data structures to store questions, multiple-choice options, and correct answers. Each line in the file represents a single question with its options and the correct answer.
- **Database Format:** Define a consistent format for storing questions, separating the question, options, and correct answer using delimiters or a structured format.
- **Read Database:** Develop functions to read and parse the question database, allowing the back-end to fetch questions for each round.
- **Write Database (Optional):** If you want to allow for dynamic addition or modification of questions, implement functions to write new questions to the database.

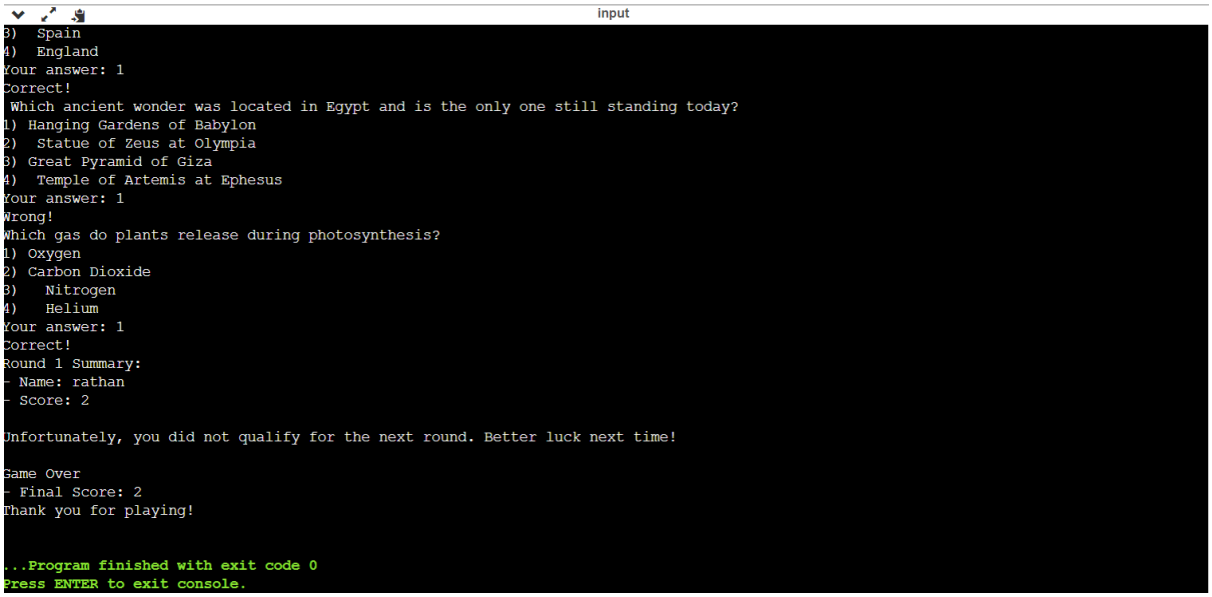
By following these design and implementation guidelines, you can create a functional Quiz Game in C programming that incorporates front-end user interaction, back-end game logic, and a simple database for question storage and retrieval.

//output


+-------+
| |
| |
| |
| // |
| // |
+--//---+
|_________|
_|_
__
/
/
/
/
/
/_______
Welcome to the Quiz Game!
Enter your name: rathan
Round 1:
What is the capital city of Brazil?
1) Rio de Janeiro
2) Sao Paulo
3) Brasilia
4) Salvador
Your answer:


input
3) Carbon Dioxide
4) Hydrogen
Your answer: 3
Correct!
What is the chemical symbol for gold?
1) Ag
2) Au
3) Fe
4) Pb
Your answer: 4
Wrong!
Who painted the Mona Lisa?
1) Vincent van Gogh
2) Leonardo da Vinci
3) Pablo Picasso
4) Michelangelo
Your answer: 1
Wrong!
Round 2 Summary:
- Name: rathan
- Score: 3
Congratulations! You've qualified for the next round.
Round 3:
In which year did the Titanic sink?
1) 1912
2) 1923
3) 1905
4) 1931
Your answer:


input
3) Spain
4) England
Your answer: 1
Correct!
Which ancient wonder was located in Egypt and is the only one still standing today?
1) Hanging Gardens of Babylon
2) Statue of Zeus at Olympia
3) Great Pyramid of Giza
4) Temple of Artemis at Ephesus
Your answer: 1
Wrong!
Which gas do plants release during photosynthesis?
1) Oxygen
2) Carbon Dioxide
3) Nitrogen
4) Helium
Your answer: 1
Correct!
Round 1 Summary:
- Name: rathan
- Score: 4
Congratulations! You've qualified for the next round.
Round 2:
What is the smallest unit of life?
1) Cell
2) Atom
3) Molecule
4) Proton
Your answer:


input
3) Spain
4) England
Your answer: 1
Correct!
Which ancient wonder was located in Egypt and is the only one still standing today?
1) Hanging Gardens of Babylon
2) Statue of Zeus at Olympia
3) Great Pyramid of Giza
4) Temple of Artemis at Ephesus
Your answer: 1
Wrong!
Which gas do plants release during photosynthesis?
1) Oxygen
2) Carbon Dioxide
3) Nitrogen
4) Helium
Your answer: 1
Correct!
Round 1 Summary:
- Name: rathan
- Score: 2
Unfortunately, you did not qualify for the next round. Better luck next time!
Game Over
- Final Score: 2
Thank you for playing!
...Program finished with exit code 0
Press ENTER to exit console.