CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF COMPUTING (UIC)

MINOR PROJECT (24BCV-1 (A))

QUIZ GAME

NAME: VISHWESH SHARMA

Subject :WEB DESIGNING

Subject code :24CAH-153

UID: 24bca20007 BCA(AR/VR) BCV-1

Submitted to: Mr Jagwinder Singh

ABSTRACT:

This project presents a web-based **Quiz Game** application designed entirely using **HTML**, **CSS**, and **JavaScript**, emphasizing an elegant **dark mode user interface**. The game engages users with a series of **multiple-choice questions** and offers **instant feedback** on their selections, tracking scores throughout the session. The project reflects the principles of interactive front-end development by implementing a clean layout, smooth transitions, and dynamic content updates.

The application is optimized for user experience through a responsive design that adapts to various screen sizes and input types. The goal is to demonstrate the effective combination of structure, style, and functionality in a single-page web application, suitable for learning, entertainment, or educational assessment purposes.

INTRODUCTION:

The primary goal of this project is to create a **functional, responsive, and visually cohesive quiz application** using modern web technologies. The application consists of a dark-themed interface for better user focus and comfort, especially in low-light environments. The project highlights the separation of concerns through HTML for structure, CSS for styling, and JavaScript for functionality.

The game consists of multiple pre-defined questions. Each question is displayed one at a time, and users can select an answer from four given options. The application provides immediate visual feedback indicating the correct and incorrect answers and keeps track of the user's score. Upon completing the quiz, a summary screen displays the final score.

OBJECTIVES:

1. **Develop an Interactive Quiz System**:
   - Create a question-based quiz system with options for each question.
   - Allow selection of answers and provide immediate visual feedback.
2. **Implement Score Tracking and Result Display**:
   - Maintain a running score.
   - Display final results upon completion of all questions.
3. **Design a Dark Mode Interface**:
   - Implement a user-friendly, dark-themed UI.
   - Enhance readability and reduce eye strain in low-light environments.
4. **Ensure Responsiveness and Accessibility**:
   - Make the layout adaptive for various devices (desktop, tablet, and mobile).
   - Ensure accessibility through proper text contrast and element spacing.

PROJECT DESIGN:

The design of the Quiz Game is structured into three main components:

1. STRUCTURE (HTML):

The layout of the quiz game is defined with semantic HTML elements:

- **Main Container (.quiz-container)**: Encapsulates the entire game UI. Styled with padding, centered alignment, and max-width constraints.

- **Question Display (#question)**: A h2 element dynamically updated to show the current question.
- **Options Area (#options)**: Contains multiple buttons representing answer choices. These are populated dynamically using JavaScript.
- **Next Button (#next-btn)**: A button that appears only after an answer is selected, allowing users to proceed.
- **Score Display (#score)**: Displays the final score once the quiz ends.

2. Styling (CSS):

The CSS brings the dark-themed UI to life:

- **Color Scheme**:
    - #121212 for background.
    - Light-colored text (#ffffff, #f0f0f0) for high contrast.
- **Component Styling**:
    - Rounded corners and subtle shadows on the quiz container.
    - Hover and active effects on option buttons.
- **Transitions and Feedback**:
    - Smooth hover transitions using transition: 0.3s;.
    - Background color changes to indicate correct (#2e7d32) or incorrect (#c62828) selections.

3. Functionality (JavaScript):

The core logic of the quiz game resides in JavaScript:

- **Data Storage**:
    - An array of objects contains all quiz questions, options, and the correct answer.
- **Dynamic Rendering**:

    - The showQuestion() function displays each question and its corresponding options.
    - Buttons are created dynamically and assigned click handlers.
- **Answer Validation**:
    - When a user selects an answer, the system checks it against the correct answer.
    - Buttons are disabled after a selection to prevent multiple inputs.
    - Correct and incorrect answers are color-coded.
- **Score Tracking**:
    - Each correct answer increments the score.
    - The score is displayed after the last question.
- **Navigation Logic**:
    - Next button advances to the next question.
    - After the last question, the final score is shown and quiz ends.

Implementation Details:

HTML Markup

```
<div class="quiz-container">

  <h2 id="question">Question text</h2>

  <div id="options"></div>

  <button id="next-btn">Next</button>

  <div id="score"></div>

</div>
```

This basic structure provides a containerized layout where dynamic elements like question text, options, and final score can be easily updated.

CSS Highlights

```
body {
  background-color: #121212;
  color: #ffffff;
  font-family: 'Segoe UI', sans-serif;
  text-align: center;
}

.quiz-container {
  background: #1e1e1e;
  padding: 30px;
  border-radius: 10px;
  box-shadow: 0 0 10px rgba(255,255,255,0.05);
  max-width: 500px;
  margin: auto;
}

.option-btn:hover {
  background-color: #444;
}
```

```
#next-btn {
  background-color: #03dac6;
  color: #000;
}
```

JavaScript Logic

```
function showQuestion() {
  const q = quizData[currentQuestion];
  questionEl.textContent = q.question;
  optionsEl.innerHTML = "";
  q.options.forEach(option => {
    const btn = document.createElement("button");
    btn.classList.add("option-btn");
    btn.textContent = option;
    btn.onclick = () => selectAnswer(btn, q.answer);
    optionsEl.appendChild(btn);
  });
}

function selectAnswer(btn, correct) {
  const options = document.querySelectorAll(".option-btn");
  options.forEach(b => b.disabled = true);
  if (btn.textContent === correct) {
    btn.style.backgroundColor = "#2e7d32";
    score++;
  } else {
    btn.style.backgroundColor = "#c62828";
    options.forEach(b => {
      if (b.textContent === correct) {
        b.style.backgroundColor = "#2e7d32";
      }
    });
  }
  nextBtn.style.display = "inline-block";
}
```

CODE:

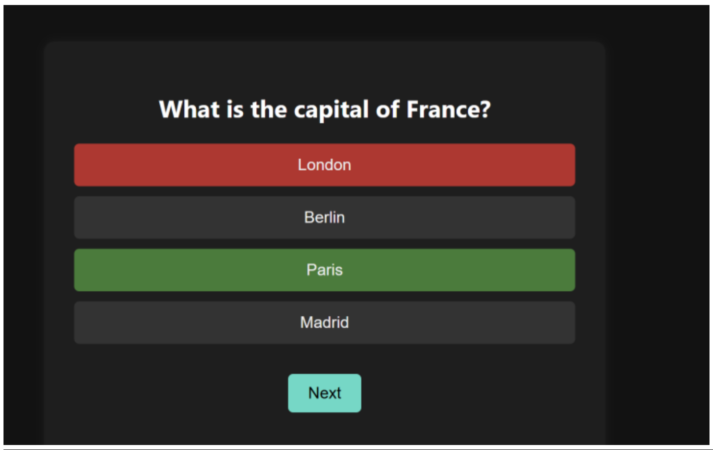
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Quiz Game - Dark Mode</title>
  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
      background-color: #121212;
      color: #ffffff;
      padding: 40px;
      text-align: center;
    }

    .quiz-container {
      max-width: 500px;
      margin: auto;
      background: #1e1e1e;
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(255,255,255,0.05);
      padding: 30px;
    }

    h2 {
      font-size: 24px;
      color: #ffffff;
    }

    .option-btn {
      display: block;
      width: 100%;
      margin: 10px 0;
      padding: 12px;
      background-color: #333;
      border: none;
      border-radius: 5px;
      font-size: 16px;
      color: #f0f0f0;
      cursor: pointer;
      transition: 0.3s;
    }

    .option-btn:hover {
      background-color: #444;
    }

    #next-btn {
      margin-top: 20px;
      padding: 10px 20px;
      font-size: 16px;
      display: none;
      background-color: #03dac6;
      color: #000;
      border: none;
      border-radius: 5px;
```

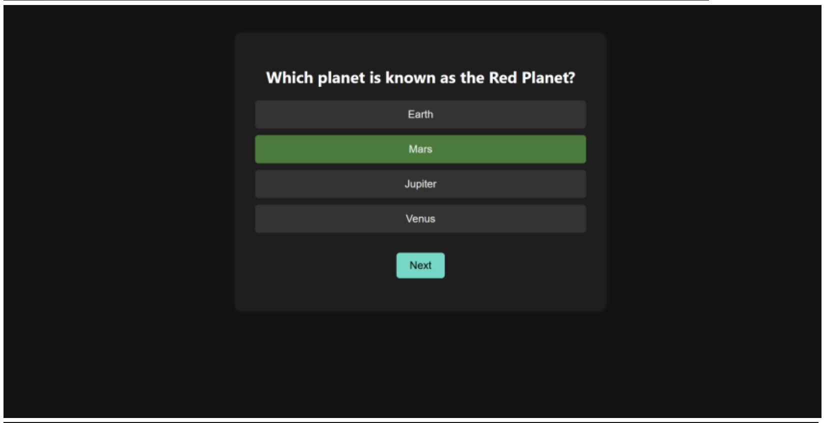
```
      cursor: pointer;
      transition: 0.3s;
    }

    #next-btn:hover {
      background-color: #00cbb0;
    }

    #score {
      font-size: 20px;
      margin-top: 20px;
      color: #ffffff;
    }
  </style>
</head>
<body>

  <div class="quiz-container">
    <h2 id="question">Question text</h2>
    <div id="options"></div>
    <button id="next-btn">Next</button>
    <div id="score"></div>
  </div>
```

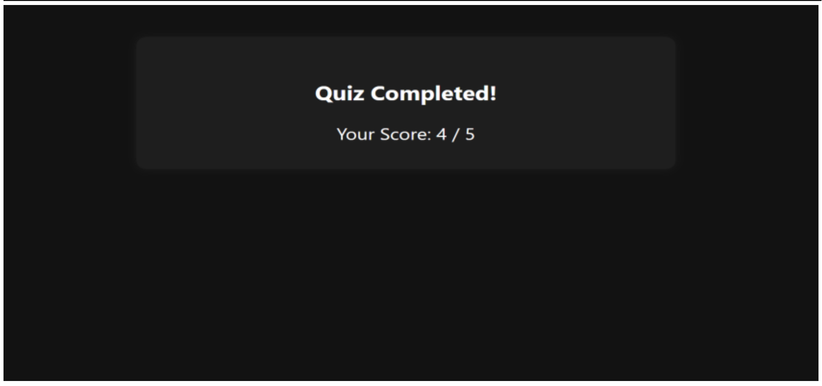
```
<script>
 const quizData = [
   {
     question: "What is the capital of France?",
     options: ["London", "Berlin", "Paris", "Madrid"],
     answer: "Paris"
   },
   {
     question: "Which planet is known as the Red Planet?",
     options: ["Earth", "Mars", "Jupiter", "Venus"],
     answer: "Mars"
   },
   {
     question: "Which language is used for web apps?",
     options: ["Python", "Java", "HTML", "C++"],
     answer: "HTML"
   },
   {
     question: "How many continents are there?",
     options: ["5", "6", "7", "8"],
     answer: "7"
   },
    {
      question: "Who wrote 'Romeo and Juliet'?",
      options: ["Charles Dickens", "William Shakespeare", "Jane Austen",
"Mark Twain"],
      answer: "William Shakespeare"
    }
  ];

  let currentQuestion = 0;
  let score = 0;

  const questionEl = document.getElementById("question");
  const optionsEl = document.getElementById("options");
  const nextBtn = document.getElementById("next-btn");
  const scoreEl = document.getElementById("score");

  function showQuestion() {
    const q = quizData[currentQuestion];
    questionEl.textContent = q.question;
    optionsEl.innerHTML = "";
    q.options.forEach(option => {
      const btn = document.createElement("button");
      btn.classList.add("option-btn");
    btn.textContent = option;
    btn.onclick = () => selectAnswer(btn, q.answer);
    optionsEl.appendChild(btn);
  });
}

function selectAnswer(btn, correct) {
  const options = document.querySelectorAll(".option-btn");
  options.forEach(b => b.disabled = true);
  if (btn.textContent === correct) {
    btn.style.backgroundColor = "#2e7d32"; // dark green
    score++;
  } else {
    btn.style.backgroundColor = "#c62828"; // red
    options.forEach(b => {
      if (b.textContent === correct) {
        b.style.backgroundColor = "#2e7d32";
      }
    });
  }
  nextBtn.style.display = "inline-block";
}
```

```
    nextBtn.onclick = () => {
      currentQuestion++;
      if (currentQuestion < quizData.length) {
        showQuestion();
        nextBtn.style.display = "none";
      } else {
        showScore();
      }
    };

    function showScore() {
      questionEl.textContent = "Quiz Completed!";
      optionsEl.innerHTML = "";
      nextBtn.style.display = "none";
      scoreEl.textContent = `Your Score: ${score} / ${quizData.length}`;
    }

    showQuestion();
  </script>

</body>
</html>
```

Execution Screenshot







Testing and Validation:

1. **Basic Functionality**:
    - All buttons are clickable and show correct feedback.
    - Next button functions correctly after selection.
2. **Score Validation**:
    - Score increments only for correct answers.
    - Final score is calculated correctly.
3. **UI Testing**:
    - Dark mode theme remains consistent across all elements.
    - Buttons respond to hover effects and visual transitions.
4. **Responsiveness**:
    - Layout remains intact across desktops, tablets, and smartphones.

Conclusion:

The **Dark Mode Quiz Game** project delivers a highly interactive and visually refined user experience using foundational web technologies. By employing responsive layout techniques, dynamic content rendering, and real-time interaction logic, the application successfully meets all outlined objectives.

It serves as a strong example of modern front-end development practices and can be enhanced further with features like question randomization, category filtering, timers, and persistent scoreboards using localStorage or back-end services.

References:

1. MDN Web Docs: https://developer.mozilla.org
2. JavaScript.info: https://javascript.info
3. FreeCodeCamp: https://www.freecodecamp.org
4. W3Schools: https://www.w3schools.com