Index

Description of Project

Introduction: The Quiz Game is a web-based project developed for the Software Development 1 course, combining knowledge testing with humor using HTML, CSS, and JavaScript.

Project Purpose: The primary purpose of the Quiz Game is to create an engaging and entertaining experience for users, combining knowledge testing with humor.

Scopes: The app is designed for web browsers and ensures compatibility across various devices, including desktops, tablets, and smartphones. It is a scalable project, allowing future enhancements such as additional questions, timers, and competitive features.

Learning Outcome: Through this project, I gained hands-on experience with:

- Structuring content with HTML.
- Styling and layout design using CSS.
- Adding interactivity with JavaScript.
- Improving project planning and execution skills.

Working Flow

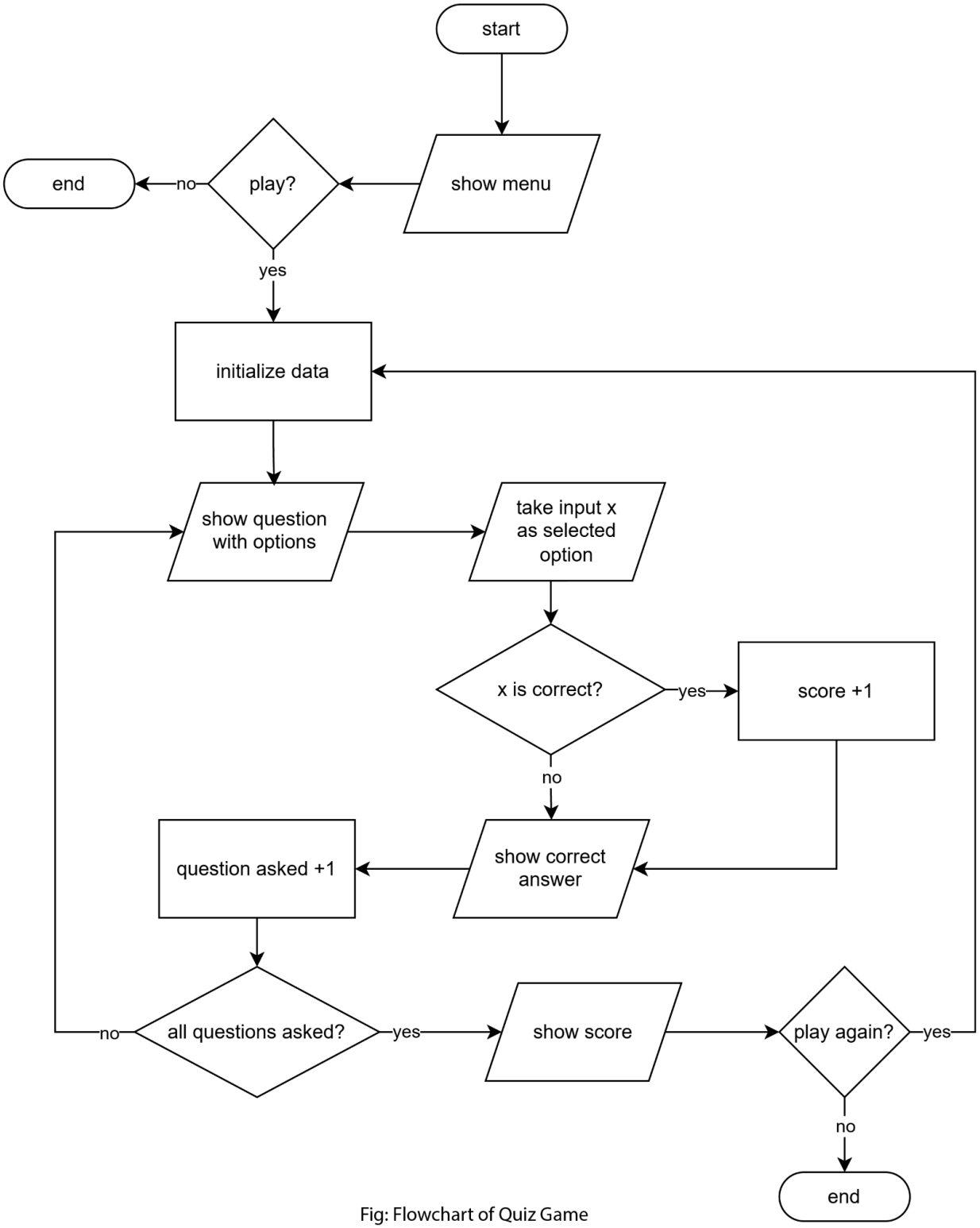


Fig: Flowchart of Quiz Game

1. Start: The game begins and transitions to the main menu.

2. Show Menu: The main menu is displayed, where the player is asked if they want to play the game.

3. Decision - Play?: The player chooses:

- If **no**, the game ends immediately.
- If **yes**, the game proceeds to initialize game data.

4. Initialize Data: The game sets up necessary data, like loading questions, resetting scores, and preparing variables to track progress.

5. Show Question with Options: A question with multiple-choice options is presented to the player.

6. Take Input as Selected Option: The player selects their answer from the options provided.

7. Decision - Is the Answer Correct?: The game checks if the selected option matches the correct answer:

- If **correct**, the score is incremented by 1.
- If **incorrect**, the correct answer is displayed.

8. Increment Question Counter: The game increments the count of questions asked so far.

9. Decision - All Questions Asked?:

- If **no**, the game loops back to the next question.
- If **yes**, the final score is displayed.

10. Show Score: The total score is shown to the player at the end of the quiz round.

11. Decision - Play Again?:

- If **yes**, the game restarts from initializing data for a new round.
- If **no**, the game ends.

12. End: The game concludes.

Code Samples

Below are code snippets showcasing key features of the Quiz Game:

HTML Structure:

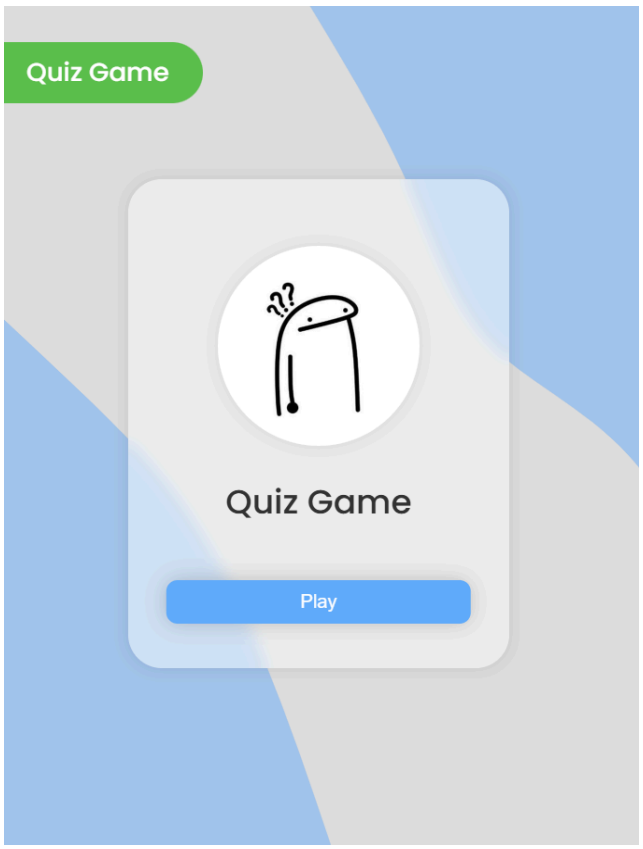
```
<div id="start" class="">
    <img src="./icon.jpg" alt="">
    <div class="t1">Quiz Game</div>
    <button onclick="startGame()">Play</button>
</div>

<div id="quiz-game" class="hide">
    <div id="count">Question 1 of 5</div>
    <div id="question">What is 2+2?</div>

    <div id="options">
        <button id="op0" class="option" onclick="buttonClicked(0)">option 1</button>
        <button id="op1" class="option" onclick="buttonClicked(1)">option 2</button>
    </div>
</div>

<div id="score" class="hide">
    <div id="msg">Better luck next time!</div>

    <div id="progress" class="progress-bar" role="progressbar" aria-valuenow="75"
    aria-valuemin="0" aria-valuemax="100">50%</div>

    <button onclick="playAgain()">Play Again</button>

</div>
```

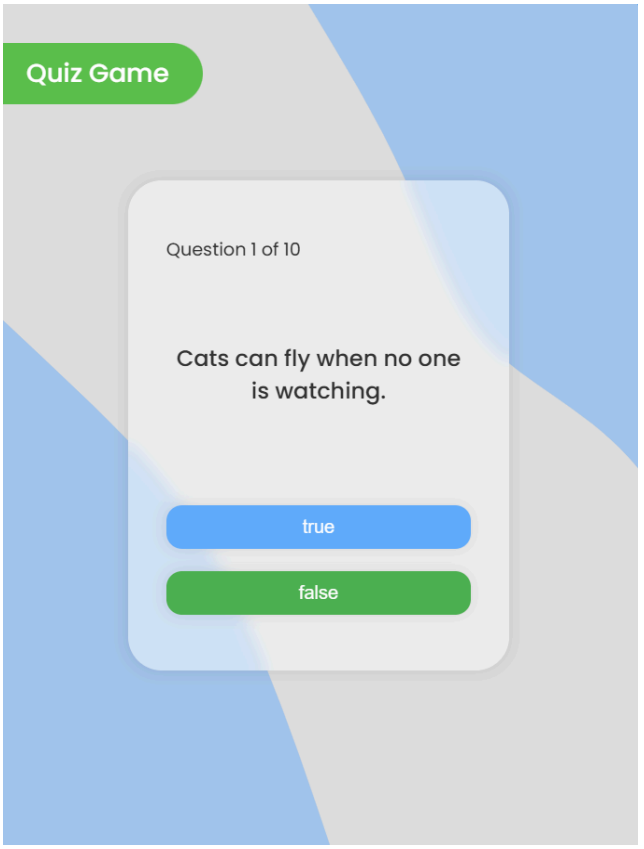
JavaScript Logic:

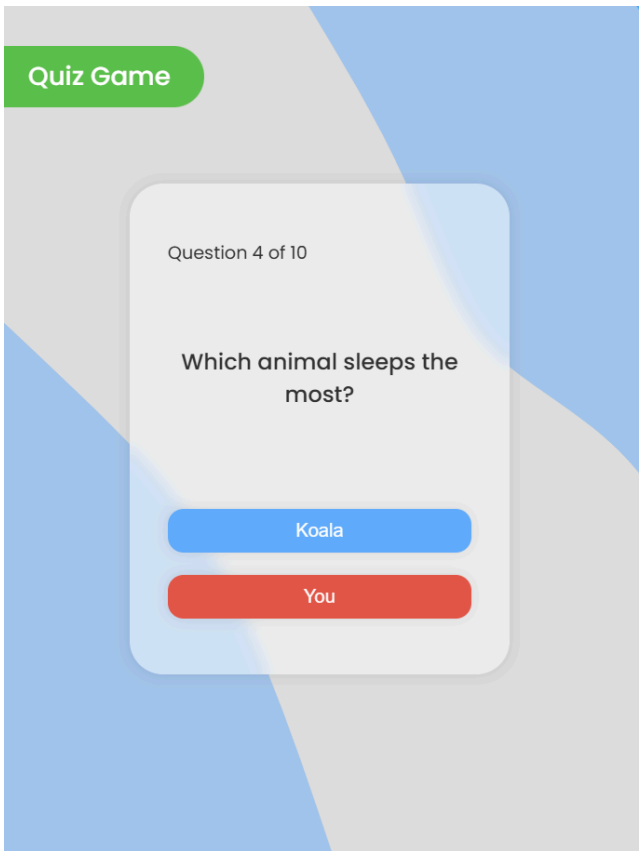
```
function displayQues() {
    if (currentQuesIndex >= maxQues) {
        showResult();
        return;
    }

    canClick = true;
    const quesData = ques[currentQuesIndex];
    const quesT = document.getElementById('question');
    const option0T = document.getElementById('op0');
    const option1T = document.getElementById('op1');


    quesT.textContent = quesData.question;
    option0T.textContent = quesData.options[0];
    option1T.textContent = quesData.options[1];

    const countT = document.getElementById('count');
    countT.textContent = `Question ${currentQuesIndex + 1} of ${maxQues}`;
}
```

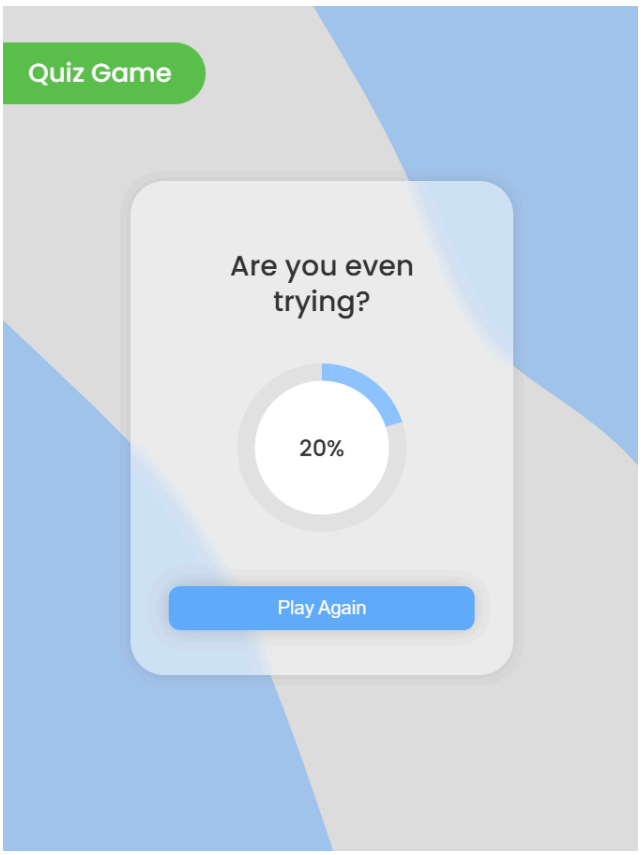
```
function buttonClicked(option) {
    if (!canClick) return;
    canClick = false;
    const quesData = ques[currentQuesIndex];
    const button = document.getElementById("op" + option);

    if (quesData.answer === option) {
        score++;
        button.classList.add('correct');
    }
    else {
        button.classList.add('wrong');
    }

    setTimeout(() => {
        button.classList.remove('correct', 'wrong');
        currentQuesIndex++;
        displayQues();
    }, 1000);
}
```

Screenshots of Runtime Output:









Future Scope

The Quiz Game has significant potential for future enhancements, including:

- **Adding More Questions:** Expanding the question bank with varied topics.
- **Leaderboard:** Implementing a ranking system to encourage competition.
- **Custom Questions:** Allowing users to add their own questions.
- **Timer Feature:** Introducing a timer for each question to add urgency.
- **Multiplayer Mode:** Enabling real-time competition between players.

Conclusion

Working on the Quiz Game has been a rewarding experience. It allowed me to consolidate my understanding of front-end development while encouraging creativity in question design. Overcoming challenges like responsive design and dynamic question loading has been particularly fulfilling. This project has strengthened my confidence in web development and inspired me to explore more interactive application ideas in the future.

Appendix

- **Code Repository:** https://github.com/lufias-69/Quiz_Game_Web
- **Playable Link:** https://saiful-anik.netlify.app/project_js/project-quiz

Scan to Play