Memory Card Game

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Memory Card Game</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <h1>Memory Card Game</h1>
    <div class="game-info">
      <div>Time: <span id="time">60</span>s</div>
      <div>Matches: <span id="matches">0</span>/8</div>
    </div>
    <div class="game-board" id="gameBoard"></div>
    <div class="message" id="message"></div>
    <div class="controls">
      <button id="restartBtn">Restart Game</button>
    </div>
  </div>
  <footer>
    <p>© 2025 Memory Card Game | Created with HTML, CSS & JavaScript | By @VipulValvaikar</p>
  </footer>
  <script src="script.js"></script>
</body>
</html>
```

style.css

```css
body {
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
    min-height: 100vh;
    display: flex;
    flex-direction: column;
    background-image:
url('https://lh3.googleusercontent.com/proxy/sDed8koYekI6qm4ybGawqHW75K8FVg1E010Le0TbeVKiBRSWzUBmeeKtRGrqpwi895WRHtuxIw79axRdhiuhHc72onFos07k22SdqQuYXi2MilGG');
    background-size: cover;
    background-position: center;
    background-attachment: fixed;
    background-color: rgba(0, 0, 0, 0.7);
    background-blend-mode: overlay;
    color: white;
}

.container {
    flex: 1;
    display: flex;
    flex-direction: column;
    align-items: center;
    padding: 20px;
}

h1 {
    color: #fff;
    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
    margin-bottom: 10px;
}

.game-info {
    display: flex;
    justify-content: space-between;
    width: 300px;
    margin-bottom: 20px;
    font-size: 18px;
    font-weight: bold;
    background-color: rgba(44, 62, 80, 0.8);
    padding: 10px 20px;
    border-radius: 10px;
}

.game-board {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    gap: 10px;
    perspective: 1000px;
    margin-bottom: 20px;
}

.card {
    width: 80px;
    height: 120px;
    position: relative;
    transform-style: preserve-3d;
    transition: transform 0.5s;
    cursor: pointer;
}

.card.flipped {
    transform: rotateY(180deg);
}

.card-face {
    position: absolute;
    width: 100%;
    height: 100%;
    backface-visibility: hidden;
    border-radius: 10px;
    display: flex;
    justify-content: center;
    align-items: center;
    font-size: 40px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
}

.card-front {
    background-color: rgba(52, 152, 219, 0.9);
    color: white;
    transform: rotateY(180deg);
}

.card-back {
    background-color: rgba(44, 62, 80, 0.9);
}

.controls {
    margin-top: 20px;
}

button {
    padding: 10px 20px;
    font-size: 16px;
    background-color: rgba(46, 204, 113, 0.9);
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s;
}

button:hover {
    background-color: rgba(39, 174, 96, 0.9);
}
```

```css
.message {
    font-size: 24px;
    font-weight: bold;
    margin-top: 20px;
    height: 30px;
    color: #fff;
    text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.8);
}

footer {
    background-color: rgba(44, 62, 80, 0.9);
    color: white;
    text-align: center;
    padding: 15px 0;
    margin-top: auto;
}

@media (max-width: 400px) {
    .game-board {
        grid-template-columns: repeat(3, 1fr);
    }
}
```

script.js

```javascript
document.addEventListener('DOMContentLoaded', () => {
    const gameBoard = document.getElementById('gameBoard');
    const timeDisplay = document.getElementById('time');
    const matchesDisplay = document.getElementById('matches');
    const messageDisplay = document.getElementById('message');
    const restartBtn = document.getElementById('restartBtn');

    const emojis = ['🐶', '🐱', '🐭', '🐹', '🐰', '🦊', '🐻', '🐼'];
    let cards = [...emojis, ...emojis];
    let flippedCards = [];
    let matchedCards = [];
    let timeLeft = 60;
    let timer;
    let gameActive = true;

    // Shuffle cards
    function shuffleCards() {
        for (let i = cards.length - 1; i > 0; i--) {
            const j = Math.floor(Math.random() * (i + 1));
            [cards[i], cards[j]] = [cards[j], cards[i]];
        }
    }

    // Create card elements
    function createCards() {
        gameBoard.innerHTML = '';
        cards.forEach((emoji, index) => {
            const card = document.createElement('div');
            card.classList.add('card');
            card.dataset.index = index;

            const cardFront = document.createElement('div');
            cardFront.classList.add('card-face', 'card-front');
            cardFront.textContent = emoji;

            const cardBack = document.createElement('div');
            cardBack.classList.add('card-face', 'card-back');

            card.appendChild(cardFront);
            card.appendChild(cardBack);
            gameBoard.appendChild(card);
        });
    }

    // Flip card
    function flipCard(card) {
        if (!gameActive || flippedCards.length >= 2 || card.classList.contains('flipped') ||
matchedCards.includes(card.dataset.index)) {
            return;
        }

        card.classList.add('flipped');
        flippedCards.push(card);

        if (flippedCards.length === 2) {
            checkForMatch();
        }
    }

    // Check for match
    function checkForMatch() {
        const [card1, card2] = flippedCards;
        const index1 = card1.dataset.index;
        const index2 = card2.dataset.index;

        if (cards[index1] === cards[index2]) {
            // Match found
            matchedCards.push(index1, index2);
            matchesDisplay.textContent = matchedCards.length / 2;
            flippedCards = [];

            // Check for win
            if (matchedCards.length === cards.length) {
                gameActive = false;
                clearInterval(timer);
                messageDisplay.textContent = 'You Win! 🎉';
            }
        } else {
            // No match
            gameActive = false;
            setTimeout(() => {
                card1.classList.remove('flipped');
                card2.classList.remove('flipped');
                flippedCards = [];
                gameActive = true;
            }, 1000);
        }
    }

    // Start timer
    function startTimer() {
        timer = setInterval(() => {
            timeLeft--;
            timeDisplay.textContent = timeLeft;

            if (timeLeft <= 0) {
                clearInterval(timer);
                gameActive = false;
                messageDisplay.textContent = 'Time\'s Up! 🥹';
                // Flip all unmatched cards
                document.querySelectorAll('.card:not(.flipped)').forEach(card => {
                    if (!matchedCards.includes(card.dataset.index)) {
                        card.classList.add('flipped');
                    }
                });
            }
        }, 1000);
    }

    // Reset game
    function resetGame() {
        flippedCards = [];
        matchedCards = [];
        timeLeft = 60;
        gameActive = true;
        timeDisplay.textContent = timeLeft;
        matchesDisplay.textContent = '0';
        messageDisplay.textContent = '';
        clearInterval(timer);
        shuffleCards();
    createCards();
    startTimer();
  }

  // Event listeners
  gameBoard.addEventListener('click', (e) => {
    const card = e.target.closest('.card');
    if (card) {
      flipCard(card);
    }
  });

  restartBtn.addEventListener('click', resetGame);

  // Initialize game
  resetGame();
});
```

What I Learned in the Process:

While building this project, I deepened my understanding of how HTML, CSS, and JavaScript work together to create an interactive experience. I learned how to:

1. Use DOM manipulation to dynamically generate cards and update game status.
2. Implement game logic such as card flipping, match detection, and win/loss conditions.
3. Use CSS transitions and transforms to animate card flipping in a visually appealing way.
4. Apply timers and intervals in JavaScript for countdown functionality.

What Challenged Me:

One of the biggest challenges was managing the card flipping logic and ensuring only two cards could be flipped at a time without breaking the game. Handling asynchronous behavior with setTimeout for unmatched cards also required careful attention to prevent bugs. Making the game responsive and visually appealing on different screen sizes with CSS was another area that required iteration.

What I'm Proud Of:

I'm proud of building a complete, polished game from scratch with separate HTML, CSS, and JS files. The card animations, background design, and overall UI look clean and engaging. I also implemented a timer-based challenge, which adds excitement and urgency to the game. Seeing everything come together into a fun, playable experience was very satisfying.

Workingcode:https://github.com/vipulvalvaikar/Memory-Card-Game-.git