Code

```
<!DOCTYPE html>
<html>
<head>
  <title>Pairs Game</title>
  <style>

    body {
      font-family: Tahoma,sans-serif;
      margin: 5px;
    }
    .title {
      font-family: Tahoma,sans-serif;
font-size: 24px;
      color: blue;
font-weight:bold;
      text-align: center;
      margin: 20px 20px 20px 20px;
    }
    .button {
      font-size: 14px;
            font-family: Tahoma,sans-serif;
padding: 5px 10px;
      margin-right: 10px;
    }
    .card {
      width: 98%;
      height: 125px;
      background-color: yellow;
      border: 1px solid #ccc;
      margin: 2px;
      display: inline-block;
      cursor: pointer;
      text-align: center;
                  font-family: Tahoma,sans-serif;
      line-height: 125px;
      font-size: 16px;
                  color: black;
    }
    .popup {
      display: none;
      position: fixed;
      top: 50%;
      left: 50%;
      transform: translate(-50%, -50%);
      background-color: #fff;
      padding: 20px;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
      z-index: 1000;
    }
.container {
 display: flex;
 align-items: center;
```

```
 justify-content: center;
padding-top: 10px;
}

.scoreboard {
                    font-family: Tahoma,sans-serif;
font-size: 16 pt;
}
    .popup-overlay {
      display: none;
      position: fixed;
      top: 0;
      left: 0;
      width: 100%;
      height: 100%;
      background-color: rgba(0,0,0,0.5);
      z-index: 900;
    }
    .pairs-table {
      width: 100%;
      border-collapse: collapse;
      margin-bottom: 5px;
    }
    .pairs-table th, .pairs-table td {
      border: 1px solid #ccc;
      padding: 5px;
      text-align: center;
    }
    .match {
      background-color: red !important;
      color: white;
    }
    .timer {
      font-size: 16px;
      font-family: Tahoma,sans-serif;
      text-align: center;
color: red;
      margin-top: 20px;
     margin-bottom: 20px;
    }
    .button: disabled {
      background-color: #ccc;
      cursor: not-allowed;
    }
  </style>
</head>
<body>
  <div class="title">Pairs Game</div>

 <div class="container"> <button class="button" id="enterPairsBtn"
onclick="enterPairs()">Enter Pairs</button>
  <button class="button" onclick="resetGame()">Reset Game</button></div>
```

```
 <div class="container"><div style="padding-right: 40px;"class="scoreboard" id="scoreboard">Score: 0</div><div style="padding-right: 40px;" class="timer" id="timer">Time Left: 100</div><div class="scoreboard" id="attempts">Pairs Attempted: 0</div></div>

<div class="container"><div style = "width: 80%"; id="gameGrid">
  <!-- Cards will be dynamically generated here -->
</div></div>

  <!-- Popup for entering pairs -->
<div class="popup-overlay" id="popupOverlay"></div>
<div class="popup" id="popup">
  <table class="pairs-table">
    <thead>
      <tr>
        <th>Pair #</th>
        <th>Card 1</th>
        <th>Card 2</th>
      </tr>
    </thead>
    <tbody id="pairsTableBody">
      <!-- Rows for entering pairs will be dynamically added here -->
    </tbody>
  </table>
  <button class="button" onclick="setupGame()">Start Game</button>
</div>

<script>
  let pairs = [];
  let cards = [];
  let flippedCards = [];
  let matchedPairs = 0;
  let attemptedPairs = 0;
  let timerInterval;
  let timeLeft = 90;

  // Function to show the popup for entering pairs
  function enterPairs() {
    document.getElementById('popupOverlay').style.display = 'block';
    document.getElementById('popup').style.display = 'block';
    document.getElementById('enterPairsBtn').disabled = true; // Disable Enter Pairs button
  }

  // Function to add rows to the pairs table in the popup
  function addPairRow() {
    const pairsTableBody = document.getElementById('pairsTableBody');
    let pairNumber = pairsTableBody.children.length + 1;
    pairsTableBody.innerHTML += `
      <tr>
        <td>${pairNumber}</td>
        <td><input type="text" id="pair${pairNumber}Card1"></td>
        <td><input type="text" id="pair${pairNumber}Card2"></td>
      </tr>
    `;
  }
```

```
    // Function to setup the game with entered pairs
    function setupGame() {
      pairs = [];
      const pairsTableBody = document.getElementById('pairsTableBody');
      pairsTableBody.querySelectorAll('tr').forEach(row => {
        const card1 =
document.getElementById(row.querySelector('input[type="text"][id$="Card1"]').id).value.trim();
        const card2 =
document.getElementById(row.querySelector('input[type="text"][id$="Card2"]').id).value.trim();
        if (card1 !== '' && card2 !== '' && card1 !== card2) {
          pairs.push({ card1, card2 });
        }
      });

      if (pairs.length === 8) {
        // Generate cards
        generateCards();
        // Close popup
        closePopup();
        // Start timer
        startTimer();
      } else {
        alert("Please enter 8 unique pairs.");
      }
    }

    // Function to generate the game grid with cards
    function generateCards() {
      // Reset variables
      matchedPairs = 0;
      attemptedPairs = 0;
      document.getElementById('scoreboard').textContent = `Score: ${matchedPairs}`;
      document.getElementById('attempts').textContent = `Pairs Attempted:
${attemptedPairs}`;

      // Clear existing cards
      document.getElementById('gameGrid').innerHTML = '';

      // Create cards array
      cards = [];
      for (let i = 0; i < 8; i++) {
        cards.push({ id: i * 2, value: pairs[i].card1, pairId: i, matched: false });
        cards.push({ id: i * 2 + 1, value: pairs[i].card2, pairId: i, matched: false });
      }

      // Shuffle cards
      cards.sort(() => Math.random() - 0.5);

      // Generate card elements
      cards.forEach(card => {
        const cardElement = document.createElement('div');
        cardElement.classList.add('card');
        cardElement.dataset.id = card.id;
```

```
            cardElement.dataset.pairId = card.pairId;
            cardElement.textContent = card.value;
            cardElement.onclick = () => flipCard(card.id);
            document.getElementById('gameGrid').appendChild(cardElement);
        });

        // Ensure the grid displays in a 4x4 layout
        document.getElementById('gameGrid').style.display = 'grid';
        document.getElementById('gameGrid').style.gridTemplateColumns = 'repeat(4, 1fr)';
        document.getElementById('gameGrid').style.gridGap = '2px';
    }

    // Function to flip a card
    function flipCard(cardId) {
        if (!cards[cardId].matched && flippedCards.length < 2 && (flippedCards.length === 0 ||
flippedCards[0].id !== cardId)) {
            // Flip the card
            const cardElement = document.querySelector(`.card[data-id="${cardId}"]`);
            cardElement.style.backgroundColor = 'white';
            flippedCards.push({ id: cardId, value: cardElement.textContent, pairId:
cardElement.dataset.pairId });

            // Check if two cards are flipped
            if (flippedCards.length === 2) {
                setTimeout(checkMatch, 500);
            }
        }
    }

    // Function to check if two flipped cards match
    function checkMatch() {
        const card1 = flippedCards[0];
        const card2 = flippedCards[1];

        attemptedPairs++;
        document.getElementById('attempts').textContent = `Pairs Attempted:
${attemptedPairs}`;

        if (card1.pairId === card2.pairId) {
            // Matched
            document.querySelector(`.card[data-id="${card1.id}"]`).classList.add('match');
            document.querySelector(`.card[data-id="${card2.id}"]`).classList.add('match');
            cards[card1.id].matched = true;
            cards[card2.id].matched = true;
            matchedPairs++;
            document.getElementById('scoreboard').textContent = `Score: ${matchedPairs}`;
            alert("You found a matching pair!");
            checkWin();
        } else {
            // Not matched
            document.querySelector(`.card[data-id="${card1.id}"]`).style.backgroundColor =
'yellow';
            document.querySelector(`.card[data-id="${card2.id}"]`).style.backgroundColor =
'yellow';
```

```
      alert("Not a match. Try again.");
    }

    // Reset flipped cards array
    flippedCards = [];
}

// Function to check if all pairs are matched
function checkWin() {
    if (matchedPairs === 8) {
        clearInterval(timerInterval);
        setTimeout(() => {
            alert("Congratulations! You've matched all pairs.");
        }, 500);
    }
}

// Function to start the timer
function startTimer() {
    clearInterval(timerInterval);
    timeLeft = 90;
    document.getElementById('timer').textContent = `Time Left: ${timeLeft}`;
    timerInterval = setInterval(() => {
        timeLeft--;
        document.getElementById('timer').textContent = `Time Left: ${timeLeft}`;
        if (timeLeft <= 0) {
            clearInterval(timerInterval);
            alert("Time's up! Game over.");
            resetGame();
        }
    }, 1000);
}

// Function to reset the game
function resetGame() {
    if (confirm("Do you want to reset the game?")) {
        clearInterval(timerInterval);
        document.getElementById('gameGrid').innerHTML = '';
        document.getElementById('scoreboard').textContent = 'Score: 0';
        document.getElementById('attempts').textContent = 'Pairs Attempted: 0';
        document.getElementById('timer').textContent = 'Time Left: 90';
        generateCards();
        startTimer(); // Restart timer after resetting game
    }
}

// Function to close the popup
function closePopup() {
    document.getElementById('popupOverlay').style.display = 'none';
    document.getElementById('popup').style.display = 'none';
}

// Add initial rows to the pairs table
for (let i = 0; i < 8; i++) {
```

```
        addPairRow();
      }
    </script>
</body>
</html>
```