Certainly! To display card images, we need to include image URLs in the card data and update the HTML and JavaScript to render those images.

### 1. \*Updated JavaScript (script.js)\*

In this updated version, each card now has an image property that stores the URL of the card's image. The card rendering functions have been modified to display these images alongside the card name, attack, and defense.

javascript

// Basic card data for player 1 and player 2, including image URLs

const player1Cards = [

{ name: 'Warrior', attack: 8, defense: 5, image: 'images/warrior.png' },

{ name: 'Mage', attack: 6, defense: 7, image: 'images/mage.png' },

{ name: 'Archer', attack: 7, defense: 6, image: 'images/archer.png' },

{ name: 'Dragon', attack: 10, defense: 8, image: 'images/dragon.png' },

{ name: 'Goblin', attack: 5, defense: 4, image: 'images/goblin.png' },

];

const player2Cards = [

{ name: 'Knight', attack: 7, defense: 7, image: 'images/knight.png' },

{ name: 'Sorcerer', attack: 6, defense: 8, image: 'images/sorcerer.png' },

{ name: 'Assassin', attack: 9, defense: 5, image: 'images/assassin.png' },

{ name: 'Giant', attack: 10, defense: 9, image: 'images/giant.png' },

{ name: 'Orc', attack: 6, defense: 5, image: 'images/orc.png' },

];

// Variables to store selected cards for both players

let selectedPlayer1Card = null;

let selectedPlayer2Card = null;

// Function to render player cards

function renderCards(player, cardArray, containerId) {

const cardContainer = document.getElementById(containerId);

cardContainer.innerHTML = ''; // Clear any existing cards

cardArray.forEach((card, index) => {

const cardDiv = document.createElement('div');

cardDiv.classList.add('card');

// Create an image element for the card

const cardImage = document.createElement('img');

cardImage.src = card.image;

cardImage.alt = card.name;

cardImage.classList.add('card-image');

// Create a title for the card

const cardTitle = document.createElement('div');

cardTitle.innerText = card.name;

cardTitle.classList.add('card-title');

// Add an event listener to handle card selection

cardDiv.addEventListener('click', () => selectCard(player, index));

// Append the image and title to the card div

cardDiv.appendChild(cardImage);

cardDiv.appendChild(cardTitle);

cardContainer.appendChild(cardDiv);

});

}

// Function to select a card for either player

function selectCard(player, index) {

if (player === 1) {

selectedPlayer1Card = player1Cards[index];

displaySelectedCard(1, selectedPlayer1Card);

} else {

selectedPlayer2Card = player2Cards[index];

displaySelectedCard(2, selectedPlayer2Card);

}

// If both players have selected their cards, resolve the battle

if (selectedPlayer1Card && selectedPlayer2Card) {

resolveBattle();

}

}

// Function to display selected card details for each player

function displaySelectedCard(player, card) {

const selectedCardDetails = document.getElementById(`selected-player${player}-card-details`);

selectedCardDetails.innerHTML = `

<h3>${card.name}</h3>

<img src="${card.image}" alt="${card.name}" class="card-image-selected">

<p>Attack: ${card.attack}</p>

<p>Defense: ${card.defense}</p>

`;

}

// Function to resolve the battle between two selected cards

function resolveBattle() {

const battleResult = document.getElementById('battle-result');

const player1Attack = selectedPlayer1Card.attack - selectedPlayer2Card.defense;

const player2Attack = selectedPlayer2Card.attack - selectedPlayer1Card.defense;

let resultMessage = '';

if (player1Attack > player2Attack) {

resultMessage = `Player 1's ${selectedPlayer1Card.name} wins!`;

} else if (player2Attack > player1Attack) {

resultMessage = `Player 2's ${selectedPlayer2Card.name} wins!`;

} else {

resultMessage = "It's a tie!";

}

// Display the result of the battle

battleResult.innerText = resultMessage;

// Reset selected cards for a new round

selectedPlayer1Card = null;

selectedPlayer2Card = null;

}

// Initialize the game by rendering the cards for both players

renderCards(1, player1Cards, 'player1-cards');

renderCards(2, player2Cards, 'player2-cards');

### 2. \*Updated HTML (index.html)\*

Make sure the image paths in the card data match the actual file paths in your project.

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Trading Card Game Battle</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<header>

<h1>Trading Card Game Battle</h1>

</header>

<main>

<!-- Player 1 Section -->

<div id="player1-hand" class="card-container">

<h2>Player 1's Hand</h2>

<div id="player1-cards"></div>

<div id="selected-player1-card">

<h2>Selected Card</h2>

<div id="selected-player1-card-details"></div>

</div>

</div>

<!-- Player 2 Section -->

<div id="player2-hand" class="card-container">

<h2>Player 2's Hand</h2>

<div id="player2-cards"></div>

<div id="selected-player2-card">

<h2>Selected Card</h2>

<div id="selected-player2-card-details"></div>

</div>

</div>

<!-- Battle Result -->

<div id="battle-result-container">

<h2>Battle Result</h2>

<p id="battle-result">Select cards to start the battle!</p>

</div>

</main>

<script src="script.js"></script>

</body>

</html>

### 3. \*Updated CSS (style.css)\*

Here is the updated CSS to style the card images and other elements.

css

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color: #f2f2f2;

}

header {

background-color: #333;

color: #fff;

text-align: center;

padding: 1rem;

}

h1 {

margin: 0;

}

main {

display: flex;

justify-content: space-around;

padding: 20px;

}

.card-container {

background-color: #fff;

padding: 1rem;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

border-radius: 8px;

}

#player1-cards, #player2-cards {

display: flex;

gap: 10px;

}

.card {

background-color: #4CAF50;

color: white;

padding: 20px;

border-radius: 8px;

text-align: center;

cursor: pointer;

transition: transform 0.2s ease;

}

.card:hover {

transform: scale(1.1);

}

.card-image {

width: 100px;

height: 150px;

object-fit: cover;

margin-bottom: 10px;

}

.card-title {

font-size: 1.2rem;

font-weight: bold;

}

#selected-player1-card, #selected-player2-card {

background-color: #fff;

padding: 1rem;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

border-radius: 8px;

width: 200px;

text-align: center;

margin-top: 10px;

}

.card-image-selected {

width: 100px;

height: 150px;

object-fit: cover;

}

#battle-result-container {

background-color: #fff;

padding: 1rem;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

border-radius: 8px;

text-align: center;

margin-top: 20px;

}

### Adding Images

Place your card images in an images/ directory (or any other folder you prefer) and ensure that the file paths in the script.js match where the images are stored. You can use placeholder images for now if you don’t have custom artwork yet.

Example structure: