ASSIGNMENT-2

Tallapaka Sathwik
192211910

QUESTION-1

HTML CODE

```
<html>
<head> <link rel="stylesheet"  href="game.css"><script src="game.js"></script>
<body>
<body>
  <div id="scoreBoard">
    Score: <span id="score">0</span> | Time Left: <span id="timer">30</span> seconds
  </div>
  <div id="gameContainer">
    <img id="movingImage" src="https://via.placeholder.com/50" alt="Moving Object">
  </div>
</body> </html>
```

CSS CODE

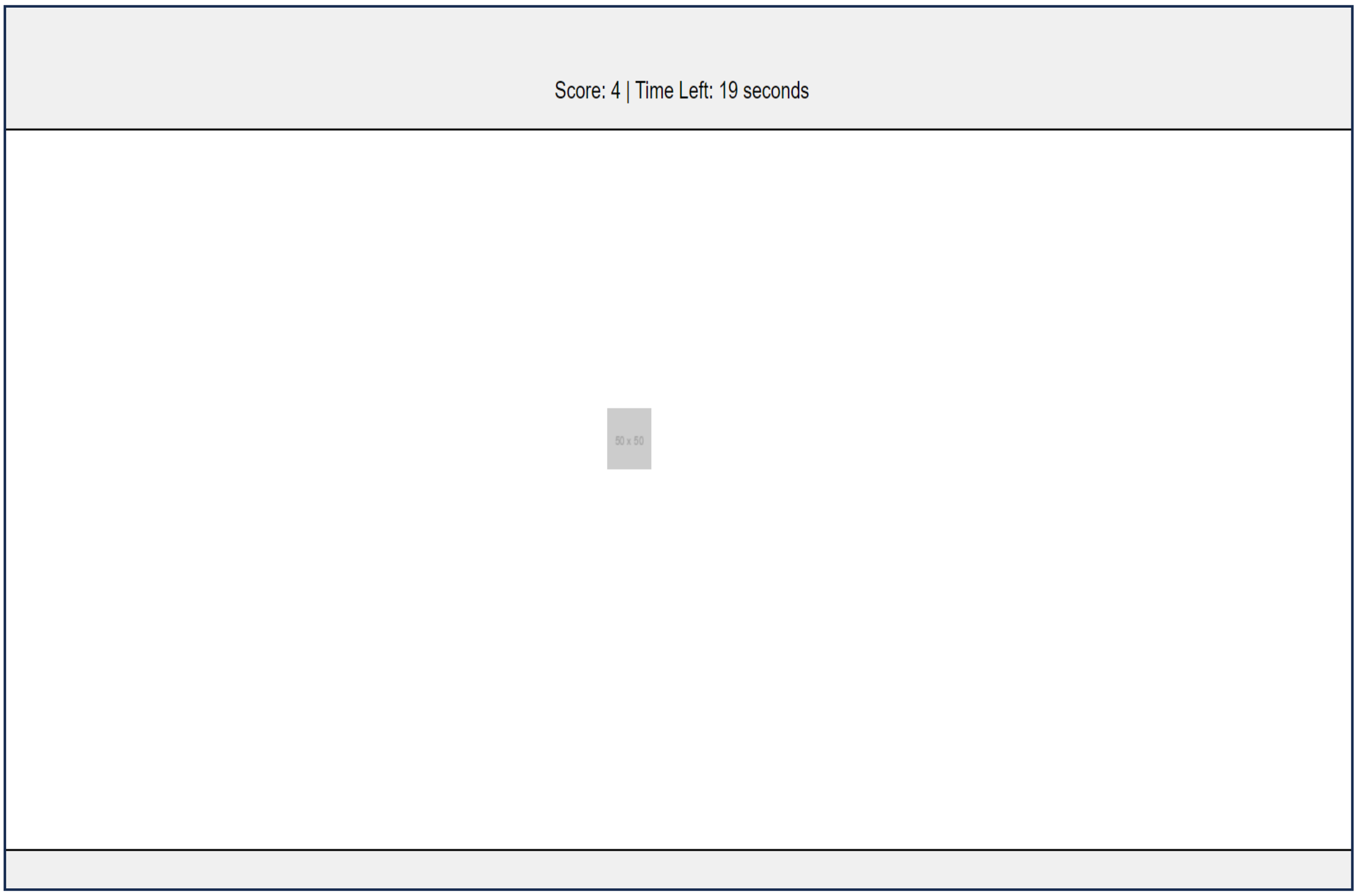
```
body {
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        display: flex;
        flex-direction: column;
        align-items: center;
        justify-content: center;
        height: 100vh;
        background-color: #f0f0f0;
        overflow: hidden;
    }
    #gameContainer {
        position: relative;
        width: 100%;
        height: 80%;
        background-color: #ffffff;
        border: 2px solid #000;
        overflow: hidden;
    }
    #movingImage {
        position: absolute;
        width: 50px;
        height: 50px;
        cursor: pointer;
    }
    #scoreBoard {
        margin: 20px;
        font-size: 20px;
    }
```

JAVASCRIPT

```
const movingImage = document.getElementById('movingImage');
    const scoreDisplay = document.getElementById('score');
    const timerDisplay = document.getElementById('timer');
    const gameContainer = document.getElementById('gameContainer');

    let score = 0;
    let timeLeft = 30;
    let timerInterval;

    function randomizePosition() {
      const containerWidth = gameContainer.offsetWidth;
      const containerHeight = gameContainer.offsetHeight;

      const newX = Math.random() * (containerWidth - movingImage.offsetWidth);
      const newY = Math.random() * (containerHeight - movingImage.offsetHeight);

      movingImage.style.left = `${newX}px`;
      movingImage.style.top = `${newY}px`;
    }
```

```
function resetGame() {
  score = 0;
  timeLeft = 30;
  scoreDisplay.textContent = score;
  timerDisplay.textContent = timeLeft;
  randomizePosition();
}

movingImage.addEventListener('click', () => {
  score++;
  scoreDisplay.textContent = score;
  randomizePosition();
});
window.onload = startGame;
```

OUTPUT


Score: 4 | Time Left: 19 seconds

QUESTION-2

HTML CODE

```
<html>
<head> <link rel=”stylesheet”  href=”task.css”><script src=”task.js”></script>
<body>
   <div id="todoApp">
      <h1>To-Do List</h1>
      <form id="taskForm">
         <input type="text" id="taskTitle" placeholder="Task Title" required>
         <textarea id="taskDescription" placeholder="Task Description" required></textarea>
         <button type="submit">Add Task</button>
      </form>
      <ul id="taskList"></ul>
   </div>
</body> </html>
```

CSS CODE

```
body {
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        display: flex;
        flex-direction: column;
        align-items: center;
        justify-content: flex-start;
        background-color: #f9f9f9;
        min-height: 100vh;
     }
     #todoApp {
        width: 100%;
        max-width: 600px;
        margin: 20px;
        background: white;
        padding: 20px;
        border-radius: 8px;
        box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
     }
```

```
#taskForm {
      display: flex;
      flex-direction: column;
      gap: 10px;
    }
    #taskList {
      margin-top: 20px;
      list-style: none;
      padding: 0;
    }
    .task {
      display: flex;
      justify-content: space-between;
      align-items: center;
      padding: 10px;
      margin-bottom: 10px;
      border: 1px solid #ddd;
      border-radius: 4px;
      background: #fff;
    }
    .task.completed {
      text-decoration: line-through;
      color: gray;
      background: #e6ffe6;
    }
    .task-buttons {
      display: flex;
      gap: 5px;
    }
    button {
      border: none;
      padding: 5px 10px;
      border-radius: 4px;
      cursor: pointer;
    }
    button.edit { background: #ffd966; }
    button.delete { background: #f28b82; }
    button.toggle { background: #c6efce; }
```

JAVASCRIPT

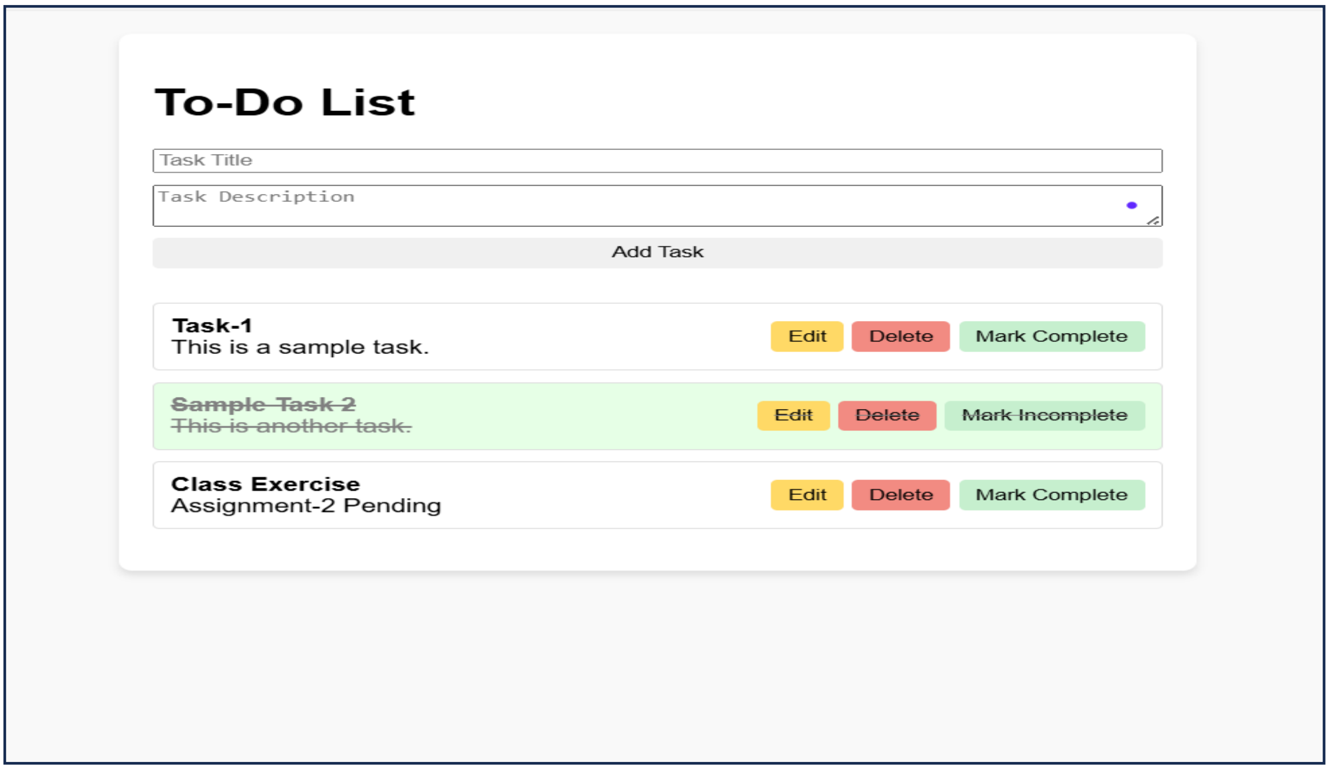
```
let tasks = [];
        document.addEventListener('DOMContentLoaded', () => {
            fetch('tasks.json')
                .then(response => response.json())
                .then(data => {
                    tasks = data;
                    renderTasks();
                });
             document.getElementById('taskForm').addEventListener('submit', addTask);
        });
        function renderTasks() {
            const taskList = document.getElementById('taskList');
            taskList.innerHTML = '';
            tasks.forEach((task, index) => {
                const taskItem = document.createElement('li');
                taskItem.className = `task ${task.status === 'complete' ? 'completed' : ''}`;
                const taskContent = document.createElement('div');
                taskContent.innerHTML = `<strong>${task.title}</strong><br>${task.description}`;
                const taskButtons = document.createElement('div');
                taskButtons.className = 'task-buttons';
                const editButton = document.createElement('button');
                editButton.className = 'edit';
                editButton.textContent = 'Edit';
                editButton.onclick = () => editTask(index);
                const deleteButton = document.createElement('button');
                deleteButton.className = 'delete';
                deleteButton.textContent = 'Delete';
                deleteButton.onclick = () => deleteTask(index);
                const toggleButton = document.createElement('button');
                toggleButton.className = 'toggle';
                toggleButton.textContent = task.status === 'complete' ? 'Mark Incomplete' : 'Mark Complete';
                toggleButton.onclick = () => toggleTaskStatus(index);
                taskButtons.append(editButton, deleteButton, toggleButton);
                taskItem.append(taskContent, taskButtons);
                taskList.appendChild(taskItem);
            });
        }
```

```
function addTask(event) {
      event.preventDefault();
      const title = document.getElementById('taskTitle').value;
      const description = document.getElementById('taskDescription').value;

      tasks.push({ title, description, status: 'incomplete' });
      renderTasks();

      document.getElementById('taskTitle').value = '';
      document.getElementById('taskDescription').value = '';
    }

    function editTask(index) {
      const task = tasks[index];
      const newTitle = prompt('Edit Task Title', task.title);
      const newDescription = prompt('Edit Task Description', task.description);

      if (newTitle !== null && newDescription !== null) {
        task.title = newTitle;
        task.description = newDescription;
        renderTasks();
      }
    }

    function deleteTask(index) {
      tasks.splice(index, 1);
      renderTasks();
    }

    function toggleTaskStatus(index) {
      const task = tasks[index];
      task.status = task.status === 'complete' ? 'incomplete' : 'complete';
      renderTasks();
    }
```

JSON

```
[
  {
    "title": "Sample Task 1",
    "description": "This is a sample task.",
    "status": "incomplete"
  },
  {
    "title": "Sample Task 2",
    "description": "This is another task.",
    "status": "complete"
  }
]
```

OUTPUT



To-Do List

Task Title

Task Description

Add Task

Task-1
This is a sample task.
Edit Delete Mark Complete

~~Sample Task 2~~
~~This is another task.~~
Edit Delete Mark Incomplete

Class Exercise
Assignment-2 Pending
Edit Delete Mark Complete