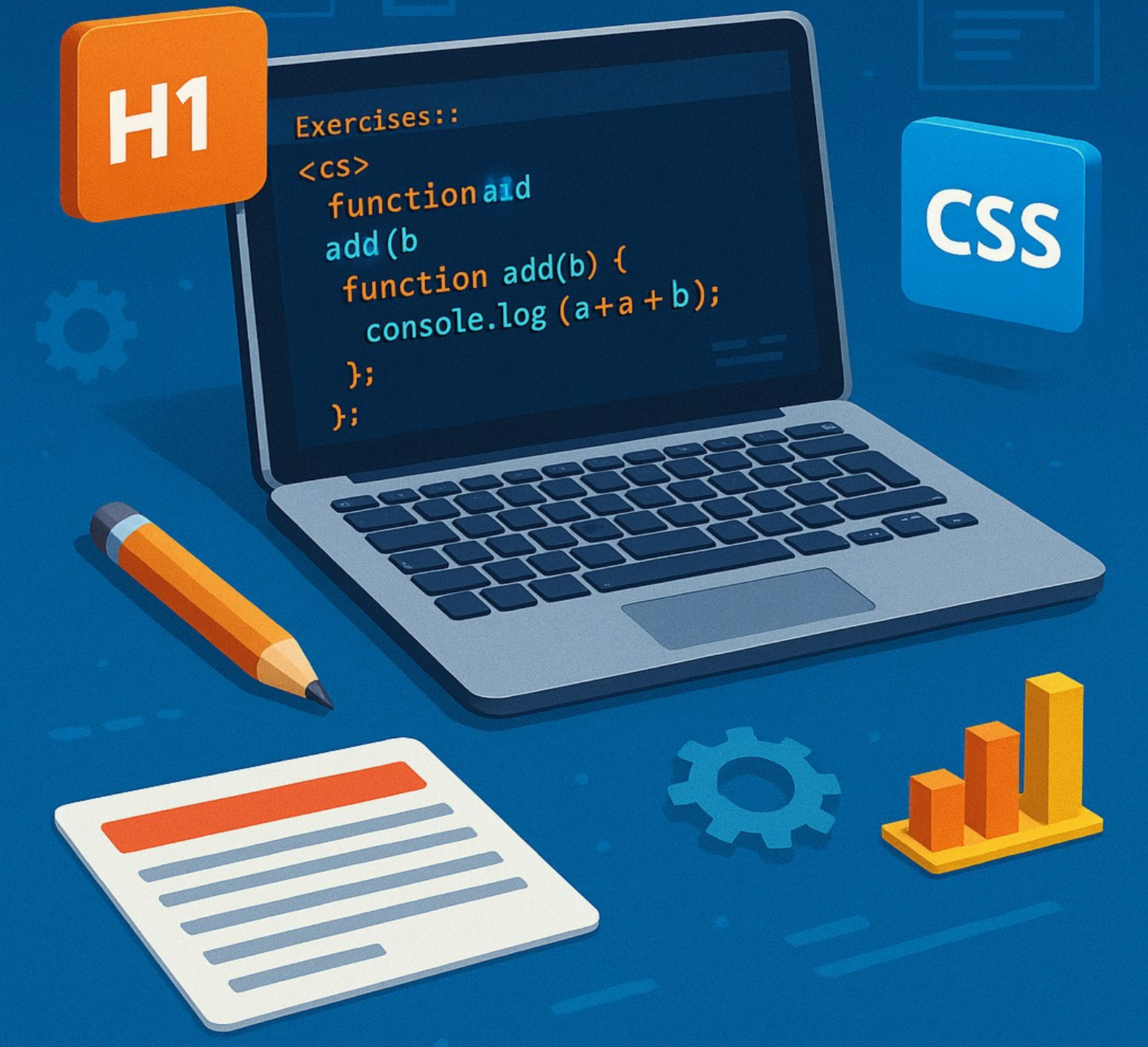
EXERCICES CORRIGÉS

HTML, CSS ET JAVASCRIPT



10 PROJETS PRATIQUES

Du débutant à l'expert

Créez des applications web complètes et fonctionnelles

📚 PLAN DES PROJETS

🟢 NIVEAU DÉBUTANT

Projet 1 : Carte de Profil Personnelle (HTML + CSS)

Projet 2 : Calculatrice Simple (HTML + CSS + JS)

Projet 3 : Compteur Interactif (HTML + CSS + JS)

🟡 NIVEAU INTERMÉDIAIRE

Projet 4 : Todo List Avancée (LocalStorage)

Projet 5 : Quiz Interactif

Projet 6 : Galerie Photos Filtrable

Projet 7 : Horloge Numérique et Analogique

🔴 NIVEAU AVANCÉ

Projet 8 : Application Météo (API)

Projet 9 : Gestionnaire de Dépenses

Projet 10 : Portfolio Complet avec Dark Mode

✅ **Code complet pour chaque projet**

🎯 Durée totale estimée : 30-40 heures

🟢 PROJET 1 — Carte de Profil Personnelle

📊 **Niveau : Débutant 🟢**

Technologies : HTML + CSS uniquement • Durée estimée : 1-2 heures • Difficulté : ⭐☆☆☆☆

🎯 **Objectifs pédagogiques**

• Structurer du contenu HTML sémantique • Utiliser Flexbox pour le centrage • Créer des effets hover avec CSS • Rendre une image ronde avec border-radius • Créer un dégradé de fond • Gérer le responsive design

📋 Description du projet

Vous allez créer une carte de profil élégante et moderne avec photo, nom, description et boutons de réseaux sociaux. C'est le projet parfait pour débuter !

🎨 Fonctionnalités

- Photo de profil ronde avec bordure colorée
- Nom et titre professionnel
- Description personnelle
- Trois boutons de réseaux sociaux (LinkedIn, GitHub, Twitter)
- Effets hover sur les boutons
- Design responsive pour mobile

💻 Code complet

📄 **index.html**

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ma Carte de Profil</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <!-- Conteneur principal de la carte -->
  <div class="card">

    <!-- Photo de profil -->
    <img src="avatar.jpg" alt="Photo de profil" class="avatar">

    <!-- Nom complet -->
    <h1 class="name">Jean Dupont</h1>

    <!-- Titre professionnel -->
```

```
      <p class="title">Développeur Web Full Stack</p>

      <!-- Description personnelle -->
      <p class="description">
        Passionné par le code et le design. J'aime créer des
        expériences web modernes et intuitives qui résolvent
        de vrais problèmes.
      </p>

      <!-- Séparateur visuel -->
      <div class="divider"></div>

      <!-- Boutons réseaux sociaux -->
      <div class="social-buttons">
        <a href="https://linkedin.com" class="btn btn-linkedin">
          LinkedIn
        </a>
        <a href="https://github.com" class="btn btn-github">
          GitHub
        </a>
        <a href="https://twitter.com" class="btn btn-twitter">
          Twitter
        </a>
      </div>

    </div>

  </body>
```

```
</html>
```

📄 **style.css**

```
/* ========================================
   RESET ET STYLES DE BASE
   ======================================== */

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  /* Dégradé de fond élégant */
  background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
  min-height: 100vh;
  /* Centrage avec Flexbox */
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 20px;
}

/* ========================================
```

```
   CARTE PRINCIPALE
   ========================================= */

.card {
  background: white;
  border-radius: 20px;
  padding: 40px;
  max-width: 400px;
  width: 100%;
  text-align: center;
  /* Ombre portée pour effet de profondeur */
  box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);
  /* Animation d'entrée */
  animation: slideIn 0.5s ease;
}

@keyframes slideIn {
  from {
    transform: translateY(50px);
    opacity: 0;
  }
  to {
    transform: translateY(0);
    opacity: 1;
  }
}

/* =========================================
```

```
   PHOTO DE PROFIL
   ========================================= */

.avatar {
  width: 150px;
  height: 150px;
  border-radius: 50%; /* Rend l'image parfaitement ronde */
  object-fit: cover; /* Garde les proportions */
  border: 5px solid #667eea;
  margin-bottom: 20px;
  /* Légère ombre pour donner du relief */
  box-shadow: 0 8px 16px rgba(102, 126, 234, 0.3);
}

/* =========================================
   TEXTES
   ========================================= */

/* Nom complet */
.name {
  font-size: 28px;
  color: #1a202c;
  margin-bottom: 10px;
  font-weight: 700;
}

/* Titre professionnel */
.title {
```

```
  font-size: 18px;
  color: #667eea;
  font-weight: 600;
  margin-bottom: 20px;
}

/* Description */
.description {
  color: #4a5568;
  line-height: 1.6;
  margin-bottom: 25px;
  font-size: 16px;
}

/* Séparateur visuel */
.divider {
  width: 60px;
  height: 3px;
  background: #667eea;
  margin: 25px auto;
  border-radius: 2px;
}

/* ======================================
   BOUTONS RÉSEAUX SOCIAUX
   ====================================== */

/* Container des boutons */
```

```
.social-buttons {
  display: flex;
  gap: 10px;
  justify-content: center;
  flex-wrap: wrap; /* Permet le passage à la ligne */
}

/* Style de base des boutons */
.btn {
  flex: 1;
  min-width: 100px;
  padding: 12px 20px;
  border-radius: 8px;
  text-decoration: none;
  font-weight: 600;
  font-size: 14px;
  /* Transition fluide pour tous les changements */
  transition: all 0.3s ease;
  display: inline-block;
}

/* Couleurs spécifiques par réseau */
.btn-linkedin {
  background: #0077b5;
  color: white;
}

.btn-github {
```

```
  background: #333;
  color: white;
}

.btn-twitter {
  background: #1da1f2;
  color: white;
}

/* Effets au survol */
.btn:hover {
  /* Légère élévation */
  transform: translateY(-3px);
  /* Ombre plus prononcée */
  box-shadow: 0 10px 20px rgba(0, 0, 0, 0.2);
}

/* Effet au clic */
.btn:active {
  transform: translateY(-1px);
}

/* ========================================
   RESPONSIVE DESIGN
   ======================================== */

/* Tablettes et petits écrans */
@media (max-width: 768px) {
  .card {
```

```
    padding: 30px;
  }

  .avatar {
    width: 120px;
    height: 120px;
  }
}

/* Smartphones */
@media (max-width: 480px) {
  body {
    padding: 10px;
  }

  .card {
    padding: 25px;
  }

  /* Boutons en colonne sur mobile */
  .social-buttons {
    flex-direction: column;
  }

  .btn {
    width: 100%;
  }
}
```

💡 💡 Astuce

Pour la photo de profil, utilisez une image carrée (500x500px minimum) pour un meilleur rendu. Les sites comme unsplash.com, pexels.com offrent des photos gratuites de qualité.

✅ 🎨 Personnalisations suggérées

1. Changez les couleurs du dégradé (ligne 12) • 2. Modifiez les couleurs des boutons • 3. Ajoutez plus de boutons (Instagram, Facebook) • 4. Utilisez vos vraies informations • 5. Ajoutez Font Awesome pour des icônes (https://fontawesome.com)

🟢 PROJET 2 — Calculatrice Simple

📊 Niveau : Débutant 🟢

Technologies : HTML + CSS + JavaScript • Durée estimée : 2-3 heures • Difficulté : ⭐⭐☆☆☆

🎯 Objectifs pédagogiques

• Créer une grille avec CSS Grid • Gérer les événements de clic • Manipuler des chaînes de caractères • Utiliser eval() pour calculer • Gérer les cas d'erreur

📋 Description du projet

Une calculatrice fonctionnelle avec les 4 opérations de base. Interface intuitive et responsive.

💻 Code complet

📄 index.html

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Calculatrice</title>
  <link rel="stylesheet" href="style.css">
</head>
```

```
<body>


  <div class="calculator">
    <!-- Écran d'affichage -->
    <input type="text" id="display" class="display" readonly>


    <!-- Boutons -->
    <div class="buttons">
      <button class="btn btn-clear"
onclick="clearDisplay()">C</button>
      <button class="btn btn-operator"
onclick="appendToDisplay('/')">/</button>
      <button class="btn btn-operator"
onclick="appendToDisplay('*')">×</button>
      <button class="btn btn-delete"
onclick="deleteLastChar()">←</button>


      <button class="btn"
onclick="appendToDisplay('7')">7</button>
      <button class="btn"
onclick="appendToDisplay('8')">8</button>
      <button class="btn"
onclick="appendToDisplay('9')">9</button>
      <button class="btn btn-operator"
onclick="appendToDisplay('-')">-</button>


      <button class="btn"
onclick="appendToDisplay('4')">4</button>
      <button class="btn"
onclick="appendToDisplay('5')">5</button>
      <button class="btn"
```

```
onclick="appendToDisplay('6')">6</button>
      <button class="btn btn-operator"
onclick="appendToDisplay('+')">+</button>

      <button class="btn"
onclick="appendToDisplay('1')">1</button>
      <button class="btn"
onclick="appendToDisplay('2')">2</button>
      <button class="btn"
onclick="appendToDisplay('3')">3</button>
      <button class="btn btn-equals" onclick="calculateResult()"
style="grid-row: span 2">=</button>

      <button class="btn btn-zero"
onclick="appendToDisplay('0')" style="grid-column: span
2">0</button>
      <button class="btn"
onclick="appendToDisplay('.')">.</button>
    </div>
  </div>

  <script src="script.js"></script>
</body>
</html>
```

📄 **style.css**

```
* {
  margin: 0;
  padding: 0;
```

```
  box-sizing: border-box;
}

body {
  font-family: Arial, sans-serif;
  background: linear-gradient(135deg, #1e3c72 0%, #2a5298 100%);
  min-height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 20px;
}

.calculator {
  background: #222;
  border-radius: 15px;
  padding: 20px;
  box-shadow: 0 20px 50px rgba(0, 0, 0, 0.5);
  max-width: 350px;
  width: 100%;
}

/* Écran */
.display {
  width: 100%;
  height: 80px;
  background: #1a1a1a;
  border: none;
```

```
  border-radius: 10px;
  color: white;
  font-size: 32px;
  text-align: right;
  padding: 20px;
  margin-bottom: 20px;
  font-family: 'Courier New', monospace;
}

/* Grille de boutons */
.buttons {
  display: grid;
  grid-template-columns: repeat(4, 1fr);
  gap: 10px;
}

/* Style de base des boutons */
.btn {
  height: 60px;
  border: none;
  border-radius: 10px;
  font-size: 20px;
  font-weight: bold;
  cursor: pointer;
  transition: all 0.2s;
  background: #4a4a4a;
  color: white;
}
```

```
.btn:hover {
  background: #5a5a5a;
  transform: scale(1.05);
}

.btn:active {
  transform: scale(0.95);
}

/* Boutons spéciaux */
.btn-operator {
  background: #ff9500;
}

.btn-operator:hover {
  background: #ffb143;
}

.btn-equals {
  background: #34c759;
}

.btn-equals:hover {
  background: #5dd879;
}

.btn-clear {
```

```
  background: #ff3b30;
}

.btn-clear:hover {
  background: #ff6259;
}

.btn-delete {
  background: #666;
}

@media (max-width: 400px) {
  .calculator {
    padding: 15px;
  }
  .btn {
    height: 50px;
    font-size: 18px;
  }
}
```

📄 **script.js**

```
// Récupérer l'élément d'affichage
const display = document.getElementById('display');

// Ajouter un caractère à l'affichage
function appendToDisplay(value) {
```

```
  display.value += value;
}

// Effacer tout l'affichage
function clearDisplay() {
  display.value = '';
}

// Supprimer le dernier caractère
function deleteLastChar() {
  display.value = display.value.slice(0, -1);
}

// Calculer le résultat
function calculateResult() {
  try {
    // eval() évalue l'expression mathématique
    display.value = eval(display.value);
  } catch (error) {
    // En cas d'erreur, afficher 'Erreur'
    display.value = 'Erreur';
    setTimeout(clearDisplay, 1500);
  }
}

// Support du clavier
document.addEventListener('keydown', (e) => {
  if (e.key >= '0' && e.key <= '9' || e.key === '.') {
```

```
    appendToDisplay(e.key);
  } else if (e.key === '+' || e.key === '-' || e.key === '*' ||
e.key === '/') {
    appendToDisplay(e.key);
  } else if (e.key === 'Enter') {
    calculateResult();
  } else if (e.key === 'Backspace') {
    deleteLastChar();
  } else if (e.key === 'Escape') {
    clearDisplay();
  }
});
```

⚠️ ⚠️ Note sur eval()

eval() est pratique mais peut être dangereux en production. Pour un vrai projet, utilisez une bibliothèque comme math.js ou créez votre propre parser.

🟢 PROJET 3 — Compteur Interactif

📊 **Niveau : Débutant 🟢**

Technologies : HTML + CSS + JavaScript • Durée estimée : 1 heure • Difficulté : ⭐☆☆☆☆

📄 **index.html**

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Compteur</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <h1>Compteur</h1>
    <div class="counter-display" id="counter">0</div>
    <div class="buttons">
      <button class="btn btn-decrease"
onclick="decrease()">-</button>
      <button class="btn btn-reset"
onclick="reset()">Reset</button>
      <button class="btn btn-increase"
onclick="increase()">+</button>
    </div>
  </div>
  <script src="script.js"></script>
```

```
</body>
</html>
```

📄 **style.css**

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: Arial, sans-serif;
  background: #f0f0f0;
  min-height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
}

.container {
  background: white;
  padding: 50px;
  border-radius: 15px;
  box-shadow: 0 10px 30px rgba(0,0,0,0.1);
  text-align: center;
}
```

```
h1 {
  color: #333;
  margin-bottom: 30px;
  font-size: 36px;
}

.counter-display {
  font-size: 120px;
  font-weight: bold;
  color: #3B82F6;
  margin: 30px 0;
  transition: all 0.3s;
}

.counter-display.positive {
  color: #10B981;
}

.counter-display.negative {
  color: #EF4444;
}

.buttons {
  display: flex;
  gap: 15px;
  justify-content: center;
}
```

```
.btn {
  padding: 15px 30px;
  font-size: 24px;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  transition: all 0.3s;
  font-weight: bold;
}

.btn-decrease {
  background: #EF4444;
  color: white;
}

.btn-reset {
  background: #6B7280;
  color: white;
}

.btn-increase {
  background: #10B981;
  color: white;
}

.btn:hover {
  transform: scale(1.1);
  box-shadow: 0 5px 15px rgba(0,0,0,0.2);
```

```
}
```

📄 **script.js**

```
let count = 0;
const counterDisplay = document.getElementById('counter');

function updateDisplay() {
  counterDisplay.textContent = count;

  // Changer la couleur selon la valeur
  counterDisplay.classList.remove('positive', 'negative');
  if (count > 0) {
    counterDisplay.classList.add('positive');
  } else if (count < 0) {
    counterDisplay.classList.add('negative');
  }
}

function increase() {
  count++;
  updateDisplay();
}

function decrease() {
  count--;
  updateDisplay();
}
```

```
function reset() {
  count = 0;
  updateDisplay();
}
```

🟡 PROJET 4 — Todo List Avancée

> 📊 **Niveau : Intermédiaire 🟡**
>
> Technologies : HTML + CSS + JavaScript + LocalStorage • Durée : 3-4 heures • Difficulté : ⭐⭐⭐☆☆

🎯 Fonctionnalités principales

- Ajouter des tâches
- Marquer comme terminées
- Supprimer des tâches
- Filtrer (Toutes / Actives / Terminées)
- Sauvegarde automatique avec LocalStorage
- Compteur de tâches restantes

📄 **index.html**

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Todo List</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <h1>📝 Ma Todo List</h1>

    <div class="todo-input">
      <input type="text" id="taskInput" placeholder="Ajouter une
```

```
tâche...">
      <button onclick="addTask()">Ajouter</button>
    </div>

    <div class="filters">
      <button class="filter-btn active"
onclick="filterTasks('all')">Toutes</button>
      <button class="filter-btn"
onclick="filterTasks('active')">Actives</button>
      <button class="filter-btn"
onclick="filterTasks('completed')">Terminées</button>
    </div>

    <ul id="taskList"></ul>

    <div class="footer">
      <span id="taskCount">0 tâche(s) restante(s)</span>
      <button onclick="clearCompleted()">Effacer
terminées</button>
    </div>
  </div>
  <script src="script.js"></script>
</body>
</html>
```

📄 **script.js**

```
let tasks = JSON.parse(localStorage.getItem('tasks')) || [];
let currentFilter = 'all';
```

```
function saveTasks() {
  localStorage.setItem('tasks', JSON.stringify(tasks));
}

function addTask() {
  const input = document.getElementById('taskInput');
  const text = input.value.trim();

  if (text === '') return;

  tasks.push({
    id: Date.now(),
    text: text,
    completed: false
  });

  input.value = '';
  saveTasks();
  renderTasks();
}

function toggleTask(id) {
  const task = tasks.find(t => t.id === id);
  if (task) {
    task.completed = !task.completed;
    saveTasks();
    renderTasks();
```

```
  }
}

function deleteTask(id) {
  tasks = tasks.filter(t => t.id !== id);
  saveTasks();
  renderTasks();
}

function filterTasks(filter) {
  currentFilter = filter;
  document.querySelectorAll('.filter-btn').forEach(btn => {
    btn.classList.remove('active');
  });
  event.target.classList.add('active');
  renderTasks();
}

function clearCompleted() {
  tasks = tasks.filter(t => !t.completed);
  saveTasks();
  renderTasks();
}

function renderTasks() {
  const taskList = document.getElementById('taskList');
  taskList.innerHTML = '';
```

```
  let filteredTasks = tasks;
  if (currentFilter === 'active') {
    filteredTasks = tasks.filter(t => !t.completed);
  } else if (currentFilter === 'completed') {
    filteredTasks = tasks.filter(t => t.completed);
  }

  filteredTasks.forEach(task => {
    const li = document.createElement('li');
    li.className = task.completed ? 'completed' : '';
    li.innerHTML = `
      <input type="checkbox" ${task.completed ? 'checked' : ''} onchange="toggleTask(${task.id})">
      <span>${task.text}</span>
      <button onclick="deleteTask(${task.id})">×</button>
    `;
    taskList.appendChild(li);
  });

  const activeCount = tasks.filter(t => !t.completed).length;
  document.getElementById('taskCount').textContent = `${activeCount} tâche(s) restante(s)`;
}

document.getElementById('taskInput').addEventListener('keypress', (e) => {
  if (e.key === 'Enter') addTask();
});
```

```
    renderTasks();
```

🟡 PROJET 5 — Quiz Interactif

📊 Niveau : Intermédiaire 🟡

Technologies : HTML + CSS + JavaScript • Durée estimée : 3-4 heures • Difficulté : ⭐⭐⭐☆☆

🎯 Objectifs pédagogiques

• Manipuler des tableaux d'objets • Gérer l'état de l'application • Afficher dynamiquement du contenu • Calculer des pourcentages • Créer une interface interactive

📋 Description du projet

Créez un quiz interactif avec 10 questions à choix multiples. Le score s'affiche en temps réel et un écran de résultat final donne le pourcentage de réussite.

🎨 Fonctionnalités

- 10 questions avec 4 choix de réponse chacune
- Affichage de la question en cours
- Feedback visuel (vert si correct, rouge si faux)
- Barre de progression
- Score en temps réel
- Écran de résultat avec pourcentage
- Bouton recommencer

💻 Code complet

📄 **index.html**

```
<!DOCTYPE html>
```

```
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Quiz Interactif</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <div class="container">
    <!-- En-tête du quiz -->
    <div class="quiz-header">
      <h1>🧠 Quiz de Culture Générale</h1>
      <div class="score-board">
        <span>Score: <span id="score">0</span>/10</span>
        <span>Question <span id="currentQuestion">1</span>/10</span>
      </div>
      <!-- Barre de progression -->
      <div class="progress-bar">
        <div id="progress" class="progress"></div>
      </div>
    </div>

    <!-- Zone de quiz -->
    <div id="quizArea" class="quiz-area">
      <h2 id="question" class="question">Question apparaîtra ici</h2>
```

```
      <div id="answers" class="answers"></div>
      <button id="nextBtn" class="next-btn" style="display: none;">Question suivante</button>
    </div>

    <!-- Écran de résultat (caché au début) -->
    <div id="resultArea" class="result-area" style="display: none;">
      <div class="result-icon" id="resultIcon">🎉</div>
      <h2 id="resultTitle">Résultat</h2>
      <p class="result-score">
        Vous avez obtenu <span id="finalScore">0</span>/10
      </p>
      <p class="result-percent" id="percentage">0%</p>
      <p id="resultMessage" class="result-message"></p>
      <button onclick="restartQuiz()" class="restart-btn">Recommencer</button>
    </div>
  </div>

  <script src="script.js"></script>
</body>
</html>
```

📄 **style.css**

```
* {
  margin: 0;
```

```
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
  min-height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 20px;
}

.container {
  background: white;
  border-radius: 20px;
  padding: 40px;
  max-width: 700px;
  width: 100%;
  box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);
}

/* En-tête */
.quiz-header h1 {
  text-align: center;
  color: #1a202c;
  margin-bottom: 20px;
```

```
}

.score-board {
  display: flex;
  justify-content: space-between;
  font-size: 18px;
  color: #4a5568;
  margin-bottom: 15px;
  font-weight: 600;
}

/* Barre de progression */
.progress-bar {
  width: 100%;
  height: 10px;
  background: #e2e8f0;
  border-radius: 10px;
  overflow: hidden;
  margin-bottom: 30px;
}

.progress {
  height: 100%;
  background: linear-gradient(90deg, #667eea 0%, #764ba2 100%);
  transition: width 0.3s ease;
  width: 0%;
}
```

```
/* Question */
.question {
  font-size: 24px;
  color: #1a202c;
  margin-bottom: 30px;
  line-height: 1.5;
}

/* Réponses */
.answers {
  display: flex;
  flex-direction: column;
  gap: 15px;
}

.answer-btn {
  background: #f7fafc;
  border: 2px solid #e2e8f0;
  padding: 20px;
  border-radius: 10px;
  font-size: 18px;
  cursor: pointer;
  transition: all 0.3s;
  text-align: left;
}

.answer-btn:hover {
  background: #edf2f7;
```

```
  border-color: #667eea;
  transform: translateX(5px);
}

.answer-btn.correct {
  background: #d1fae5;
  border-color: #10b981;
  color: #065f46;
}

.answer-btn.incorrect {
  background: #fee2e2;
  border-color: #ef4444;
  color: #991b1b;
}

.answer-btn:disabled {
  cursor: not-allowed;
}

/* Bouton suivant */
.next-btn {
  width: 100%;
  padding: 15px;
  background: #667eea;
  color: white;
  border: none;
  border-radius: 10px;
```

```
  font-size: 18px;
  font-weight: 600;
  cursor: pointer;
  margin-top: 20px;
  transition: all 0.3s;
}

.next-btn:hover {
  background: #5568d3;
  transform: translateY(-2px);
}

/* Résultat */
.result-area {
  text-align: center;
}

.result-icon {
  font-size: 80px;
  margin-bottom: 20px;
}

#resultTitle {
  font-size: 32px;
  color: #1a202c;
  margin-bottom: 20px;
}

.result-score {
```

```
  font-size: 24px;
  color: #4a5568;
  margin-bottom: 10px;
}

.result-percent {
  font-size: 48px;
  font-weight: bold;
  color: #667eea;
  margin-bottom: 20px;
}

.result-message {
  font-size: 20px;
  color: #718096;
  margin-bottom: 30px;
}

.restart-btn {
  padding: 15px 40px;
  background: #667eea;
  color: white;
  border: none;
  border-radius: 10px;
  font-size: 18px;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s;
```

```
}

.restart-btn:hover {
  background: #5568d3;
  transform: scale(1.05);
}

@media (max-width: 600px) {
  .container {
    padding: 20px;
  }
  .question {
    font-size: 20px;
  }
  .answer-btn {
    font-size: 16px;
    padding: 15px;
  }
}
```

📄 **script.js**

```
// Base de données des questions
const questions = [
  {
    question: "Quelle est la capitale de la France ?",
    answers: ["Paris", "Lyon", "Marseille", "Bordeaux"],
```

```
    correct: 0
  },
  {
    question: "Combien font 15 × 7 ?",
    answers: ["95", "105", "115", "125"],
    correct: 1
  },
  {
    question: "Qui a peint la Joconde ?",
    answers: ["Picasso", "Van Gogh", "Léonard de Vinci",
"Michel-Ange"],
    correct: 2
  },
  {
    question: "Quel est le plus grand océan du monde ?",
    answers: ["Atlantique", "Indien", "Arctique", "Pacifique"],
    correct: 3
  },
  {
    question: "En quelle année l'homme a-t-il marché sur la Lune
?",
    answers: ["1965", "1969", "1972", "1975"],
    correct: 1
  },
  {
    question: "Combien y a-t-il de continents sur Terre ?",
    answers: ["5", "6", "7", "8"],
    correct: 2
  },
```

```
  {
    question: "Quel est le langage de programmation créé par Brendan Eich ?",
    answers: ["Python", "Java", "JavaScript", "C++"],
    correct: 2
  },
  {
    question: "Quelle est la vitesse de la lumière ?",
    answers: ["300 000 km/s", "150 000 km/s", "500 000 km/s", "1 000 000 km/s"],
    correct: 0
  },
  {
    question: "Qui a écrit 'Les Misérables' ?",
    answers: ["Émile Zola", "Victor Hugo", "Balzac", "Flaubert"],
    correct: 1
  },
  {
    question: "Combien de touches a un piano standard ?",
    answers: ["76", "88", "92", "100"],
    correct: 1
  }
];

// Variables globales
let currentQuestionIndex = 0;
let score = 0;
let hasAnswered = false;
```

```
// Éléments du DOM
const questionEl = document.getElementById('question');
const answersEl = document.getElementById('answers');
const nextBtn = document.getElementById('nextBtn');
const scoreEl = document.getElementById('score');
const currentQuestionEl =
document.getElementById('currentQuestion');
const progressEl = document.getElementById('progress');
const quizArea = document.getElementById('quizArea');
const resultArea = document.getElementById('resultArea');

// Initialiser le quiz
function initQuiz() {
  currentQuestionIndex = 0;
  score = 0;
  hasAnswered = false;
  updateScore();
  displayQuestion();
}

// Afficher la question actuelle
function displayQuestion() {
  const question = questions[currentQuestionIndex];
  hasAnswered = false;

  // Afficher la question
  questionEl.textContent = question.question;
```

```
  // Mise à jour du numéro de question
  currentQuestionEl.textContent = currentQuestionIndex + 1;

  // Mise à jour de la barre de progression
  const progress = ((currentQuestionIndex + 1) /
questions.length) * 100;
  progressEl.style.width = progress + '%';

  // Afficher les réponses
  answersEl.innerHTML = '';
  question.answers.forEach((answer, index) => {
    const btn = document.createElement('button');
    btn.className = 'answer-btn';
    btn.textContent = answer;
    btn.onclick = () => selectAnswer(index);
    answersEl.appendChild(btn);
  });

  // Cacher le bouton suivant
  nextBtn.style.display = 'none';
}

// Sélectionner une réponse
function selectAnswer(selectedIndex) {
  if (hasAnswered) return; // Empêcher de répondre plusieurs
fois

  hasAnswered = true;
```

```
  const question = questions[currentQuestionIndex];
  const buttons = answersEl.querySelectorAll('.answer-btn');

  // Vérifier si la réponse est correcte
  if (selectedIndex === question.correct) {
    score++;
    buttons[selectedIndex].classList.add('correct');
    updateScore();
  } else {
    buttons[selectedIndex].classList.add('incorrect');
    buttons[question.correct].classList.add('correct');
  }

  // Désactiver tous les boutons
  buttons.forEach(btn => btn.disabled = true);

  // Afficher le bouton suivant
  nextBtn.style.display = 'block';
}

// Passer à la question suivante
nextBtn.addEventListener('click', () => {
  currentQuestionIndex++;

  if (currentQuestionIndex < questions.length) {
    displayQuestion();
  } else {
    showResults();
```

```
  }
});

// Mettre à jour le score
function updateScore() {
  scoreEl.textContent = score;
}

// Afficher les résultats
function showResults() {
  quizArea.style.display = 'none';
  resultArea.style.display = 'block';

  const percentage = Math.round((score / questions.length) *
100);

  document.getElementById('finalScore').textContent = score;
  document.getElementById('percentage').textContent = percentage
+ '%';

  // Message selon le score
  const resultIcon = document.getElementById('resultIcon');
  const resultMessage =
document.getElementById('resultMessage');

  if (percentage === 100) {
    resultIcon.textContent = '🏆';
    resultMessage.textContent = 'Parfait ! Vous êtes un
expert !';
```

```
  } else if (percentage >= 70) {
    resultIcon.textContent = '🎉';
    resultMessage.textContent = 'Excellent travail ! Très bon score !';
  } else if (percentage >= 50) {
    resultIcon.textContent = '👍';
    resultMessage.textContent = 'Bien joué ! Vous pouvez faire mieux !';
  } else {
    resultIcon.textContent = '😅';
    resultMessage.textContent = 'Pas mal ! Réessayez pour améliorer !';
  }
}

// Recommencer le quiz
function restartQuiz() {
  quizArea.style.display = 'block';
  resultArea.style.display = 'none';
  initQuiz();
}

// Démarrer le quiz au chargement
initQuiz();
```

💡 Améliorations possibles

• Ajouter un timer pour chaque question • Mélanger les questions et réponses • Ajouter des catégories de quiz • Enregistrer le meilleur score dans LocalStorage •

Ajouter des effets sonores • Créer un mode multijoueur

🟡 PROJET 6 — Galerie Photos Filtrable

📊 Niveau : Intermédiaire 🟡

Technologies : HTML + CSS Grid + JavaScript • Durée : 3-4 heures • Difficulté : ⭐⭐⭐☆☆

🎯 Objectifs pédagogiques

• Utiliser CSS Grid pour layouts • Filtrer des éléments DOM • Créer une modal interactive • Gérer la navigation clavier • Animer des transitions

🎨 Fonctionnalités

- Grille d'images responsive
- Filtres par catégorie (Tout, Nature, Ville, Animaux)
- Animation d'apparition
- Modal pour agrandir les images
- Navigation dans la modal (flèches)

📄 index.html

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Galerie Photos</title>
  <link rel="stylesheet" href="style.css">
</head>
```

```
<body>


  <div class="container">
    <h1>📷 Ma Galerie Photos</h1>


    <!-- Filtres -->
    <div class="filters">
      <button class="filter-btn active" data-
filter="all">Tout</button>
      <button class="filter-btn"
data-filter="nature">Nature</button>
      <button class="filter-btn"
data-filter="ville">Ville</button>
      <button class="filter-btn" data-
filter="animaux">Animaux</button>
    </div>


    <!-- Grille d'images -->
    <div class="gallery" id="gallery"></div>
  </div>


  <!-- Modal pour agrandir -->
  <div class="modal" id="modal">
    <span class="close" id="closeModal">&times;</span>
    <button class="nav-btn prev" id="prevBtn">❮</button>
    <img src="" alt="" id="modalImg">
    <button class="nav-btn next" id="nextBtn">❯</button>
    <p class="caption" id="caption"></p>
  </div>
```

```
  <script src="script.js"></script>
</body>
</html>
```

📄 **style.css**

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: Arial, sans-serif;
  background: #f5f5f5;
  padding: 20px;
}

.container {
  max-width: 1200px;
  margin: 0 auto;
}

h1 {
  text-align: center;
  color: #1a202c;
  margin-bottom: 30px;
```

```
  font-size: 36px;
}

/* Filtres */
.filters {
  display: flex;
  justify-content: center;
  gap: 15px;
  margin-bottom: 40px;
  flex-wrap: wrap;
}

.filter-btn {
  padding: 12px 30px;
  background: white;
  border: 2px solid #e2e8f0;
  border-radius: 25px;
  font-size: 16px;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s;
}

.filter-btn:hover {
  border-color: #667eea;
  color: #667eea;
}
```

```
.filter-btn.active {
  background: #667eea;
  color: white;
  border-color: #667eea;
}

/* Grille d'images */
.gallery {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
  gap: 20px;
}

.gallery-item {
  position: relative;
  overflow: hidden;
  border-radius: 15px;
  cursor: pointer;
  box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  transition: transform 0.3s;
  animation: fadeIn 0.5s;
}

@keyframes fadeIn {
  from {
    opacity: 0;
    transform: scale(0.8);
  }
```

```
  to {
    opacity: 1;
    transform: scale(1);
  }
}

.gallery-item:hover {
  transform: scale(1.05);
}

.gallery-item img {
  width: 100%;
  height: 250px;
  object-fit: cover;
  display: block;
}

.gallery-item .overlay {
  position: absolute;
  bottom: 0;
  left: 0;
  right: 0;
  background: linear-gradient(transparent, rgba(0,0,0,0.8));
  padding: 20px;
  color: white;
  transform: translateY(100%);
  transition: transform 0.3s;
}
```

```
.gallery-item:hover .overlay {
  transform: translateY(0);
}

.gallery-item.hide {
  display: none;
}

/* Modal */
.modal {
  display: none;
  position: fixed;
  z-index: 1000;
  left: 0;
  top: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.95);
  align-items: center;
  justify-content: center;
}

.modal.show {
  display: flex;
}

.modal img {
  max-width: 90%;
```

```
  max-height: 80vh;
  object-fit: contain;
}

.close {
  position: absolute;
  top: 20px;
  right: 40px;
  color: white;
  font-size: 40px;
  cursor: pointer;
  z-index: 1001;
}

.nav-btn {
  position: absolute;
  top: 50%;
  transform: translateY(-50%);
  background: rgba(255,255,255,0.2);
  color: white;
  border: none;
  font-size: 30px;
  padding: 15px 20px;
  cursor: pointer;
  transition: background 0.3s;
  z-index: 1001;
}
```

```
.nav-btn:hover {
  background: rgba(255,255,255,0.4);
}

.prev {
  left: 20px;
}

.next {
  right: 20px;
}

.caption {
  position: absolute;
  bottom: 20px;
  color: white;
  font-size: 18px;
  text-align: center;
  width: 100%;
}

@media (max-width: 768px) {
  .gallery {
    grid-template-columns: repeat(auto-fill, minmax(200px,
1fr));
    gap: 10px;
  }
}
```

📄 **script.js**

```
// Base de données d'images
const images = [
  { src: 'https://picsum.photos/400/300?random=1', category: 'nature', caption: 'Paysage de montagne' },
  { src: 'https://picsum.photos/400/300?random=2', category: 'ville', caption: 'Architecture urbaine' },
  { src: 'https://picsum.photos/400/300?random=3', category: 'animaux', caption: 'Faune sauvage' },
  { src: 'https://picsum.photos/400/300?random=4', category: 'nature', caption: 'Forêt luxuriante' },
  { src: 'https://picsum.photos/400/300?random=5', category: 'ville', caption: 'Gratte-ciels' },
  { src: 'https://picsum.photos/400/300?random=6', category: 'animaux', caption: 'Animaux domestiques' },
  { src: 'https://picsum.photos/400/300?random=7', category: 'nature', caption: 'Coucher de soleil' },
  { src: 'https://picsum.photos/400/300?random=8', category: 'ville', caption: 'Rue piétonne' },
  { src: 'https://picsum.photos/400/300?random=9', category: 'animaux', caption: 'Oiseaux exotiques' },
  { src: 'https://picsum.photos/400/300?random=10', category: 'nature', caption: 'Cascade' },
  { src: 'https://picsum.photos/400/300?random=11', category: 'ville', caption: 'Pont historique' },
  { src: 'https://picsum.photos/400/300?random=12', category: 'animaux', caption: 'Vie marine' }
];


let currentImageIndex = 0;
```

```
let filteredImages = [...images];

// Éléments du DOM
const gallery = document.getElementById('gallery');
const modal = document.getElementById('modal');
const modalImg = document.getElementById('modalImg');
const caption = document.getElementById('caption');
const closeModal = document.getElementById('closeModal');
const prevBtn = document.getElementById('prevBtn');
const nextBtn = document.getElementById('nextBtn');

// Afficher la galerie
function displayGallery() {
  gallery.innerHTML = '';

  filteredImages.forEach((image, index) => {
    const item = document.createElement('div');
    item.className = 'gallery-item';
    item.innerHTML = `
      <img src="${image.src}" alt="${image.caption}">
      <div class="overlay">
        <h3>${image.caption}</h3>
        <p>${image.category}</p>
      </div>
    `;

    item.addEventListener('click', () => openModal(index));
    gallery.appendChild(item);
```

```
  });
}

// Filtrer les images
const filterBtns = document.querySelectorAll('.filter-btn');
filterBtns.forEach(btn => {
  btn.addEventListener('click', () => {
    // Retirer la classe active de tous les boutons
    filterBtns.forEach(b => b.classList.remove('active'));
    btn.classList.add('active');

    const filter = btn.dataset.filter;

    if (filter === 'all') {
      filteredImages = [...images];
    } else {
      filteredImages = images.filter(img => img.category === filter);
    }

    displayGallery();
  });
});

// Ouvrir la modal
function openModal(index) {
  currentImageIndex = index;
  updateModalImage();
```

```
  modal.classList.add('show');
}

// Fermer la modal
closeModal.addEventListener('click', () => {
  modal.classList.remove('show');
});

// Fermer en cliquant sur le fond
modal.addEventListener('click', (e) => {
  if (e.target === modal) {
    modal.classList.remove('show');
  }
});

// Navigation dans la modal
prevBtn.addEventListener('click', () => {
  currentImageIndex = (currentImageIndex - 1 +
filteredImages.length) % filteredImages.length;
  updateModalImage();
});

nextBtn.addEventListener('click', () => {
  currentImageIndex = (currentImageIndex + 1) %
filteredImages.length;
  updateModalImage();
});

// Mettre à jour l'image de la modal
```

```
function updateModalImage() {
  const image = filteredImages[currentImageIndex];
  modalImg.src = image.src;
  caption.textContent = image.caption;
}

// Support clavier
document.addEventListener('keydown', (e) => {
  if (!modal.classList.contains('show')) return;

  if (e.key === 'ArrowLeft') {
    prevBtn.click();
  } else if (e.key === 'ArrowRight') {
    nextBtn.click();
  } else if (e.key === 'Escape') {
    modal.classList.remove('show');
  }
});

// Initialiser
displayGallery();
```

> 💡 **Note**
>
> Les images utilisent picsum.photos pour des images aléatoires. Remplacez-les par vos propres images dans un vrai projet.

🟡 PROJET 7 — Horloge Numérique et Analogique

> 📊 **Niveau : Intermédiaire 🟡**
>
> Technologies : HTML + CSS + JavaScript + setInterval • Durée : 2-3 heures • Difficulté : ⭐⭐⭐☆☆

🎨 Fonctionnalités

- Affichage digital de l'heure
- Horloge analogique avec aiguilles
- Mode 12h/24h
- Mode jour/nuit

📄 index.html

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Horloge</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <h1>⏰ Horloge Mondiale</h1>

    <div class="controls">
      <button onclick="toggleFormat()">12h/24h</button>
      <button onclick="toggleTheme()">🌙 Nuit</button>
    </div>

    <div class="digital-clock" id="digitalClock">00:00:00</div>

    <div class="analog-clock">
      <div class="clock-face">
        <div class="hand hour-hand" id="hourHand"></div>
```

```
        <div class="hand minute-hand" id="minuteHand"></div>
        <div class="hand second-hand" id="secondHand"></div>
        <div class="center-dot"></div>
        <div class="number" style="--i:1"><span>1</span></div>
        <div class="number" style="--i:2"><span>2</span></div>
        <div class="number" style="--i:3"><span>3</span></div>
        <div class="number" style="--i:4"><span>4</span></div>
        <div class="number" style="--i:5"><span>5</span></div>
        <div class="number" style="--i:6"><span>6</span></div>
        <div class="number" style="--i:7"><span>7</span></div>
        <div class="number" style="--i:8"><span>8</span></div>
        <div class="number" style="--i:9"><span>9</span></div>
        <div class="number" style="--i:10"><span>10</span></div>
        <div class="number" style="--i:11"><span>11</span></div>
        <div class="number" style="--i:12"><span>12</span></div>
      </div>
    </div>
  </div>
  <script src="script.js"></script>
</body>
</html>
```

📄 **script.js**

```
let is24HourFormat = true;

function updateClock() {
  const now = new Date();
  let hours = now.getHours();
  const minutes = now.getMinutes();
  const seconds = now.getSeconds();

  // Digital clock
  const digitalClock = document.getElementById('digitalClock');
  if (is24HourFormat) {
    digitalClock.textContent =
      `${String(hours).padStart(2, '0')}:$
{String(minutes).padStart(2, '0')}:${String(seconds).padStart(2,
```

```
'0')}`;
  } else {
    const period = hours >= 12 ? 'PM' : 'AM';
    hours = hours % 12 || 12;
    digitalClock.textContent =
      `${String(hours).padStart(2, '0')}:$
{String(minutes).padStart(2, '0')}:${String(seconds).padStart(2,
'0')} ${period}`;
  }

  // Analog clock
  const hourHand = document.getElementById('hourHand');
  const minuteHand = document.getElementById('minuteHand');
  const secondHand = document.getElementById('secondHand');

  const hourDeg = (hours % 12) * 30 + minutes * 0.5;
  const minuteDeg = minutes * 6 + seconds * 0.1;
  const secondDeg = seconds * 6;

  hourHand.style.transform = `rotate(${hourDeg}deg)`;
  minuteHand.style.transform = `rotate(${minuteDeg}deg)`;
  secondHand.style.transform = `rotate(${secondDeg}deg)`;
}

function toggleFormat() {
  is24HourFormat = !is24HourFormat;
}

function toggleTheme() {
  document.body.classList.toggle('dark');
}

setInterval(updateClock, 1000);
updateClock();
```

🔴 PROJET 8 — Application Météo (API)

📊 Niveau : Avancé 🔴

Technologies : HTML + CSS + JavaScript + Fetch API • Durée : 4-5 heures • Difficulté : ⭐⭐⭐⭐☆

🌤️ Setup API

1. Créez un compte gratuit sur openweathermap.org

2. Obtenez votre clé API

3. Utilisez l'API : api.openweathermap.org/data/2.5/weather

📄 script.js (extrait clé)

```
const API_KEY = 'VOTRE_CLE_API_ICI';
const API_URL = 'https://api.openweathermap.org/data/2.5/weather';

async function getWeather(city) {
  try {
    const response = await fetch(
      `${API_URL}?q=${city}&appid=${API_KEY}&units=metric&lang=fr`
    );

    if (!response.ok) throw new Error('Ville non trouvée');

    const data = await response.json();
    displayWeather(data);
  } catch (error) {
    alert(error.message);
  }
}

function displayWeather(data) {
  document.getElementById('cityName').textContent = data.name;
  document.getElementById('temperature').textContent =
```

```
    Math.round(data.main.temp) + '°C';
  document.getElementById('description').textContent =
    data.weather[0].description;
  document.getElementById('humidity').textContent =
    data.main.humidity + '%';
  document.getElementById('wind').textContent =
    data.wind.speed + ' km/h';

  // Icône météo
  const icon = data.weather[0].icon;
  document.getElementById('weatherIcon').src =
    `http://openweathermap.org/img/wn/${icon}@2x.png`;
}

// Recherche
document.getElementById('searchBtn').addEventListener('click', ()
=> {
  const city = document.getElementById('cityInput').value;
  if (city) getWeather(city);
});
```

🔴 PROJET 9 — Gestionnaire de Dépenses

📊 **Niveau : Avancé 🔴**

Technologies : HTML + CSS + JS + LocalStorage + Chart.js • Durée : 5-6 heures • Difficulté : ⭐⭐⭐⭐☆

📄 **script.js (logique principale)**

```
let transactions = JSON.parse(localStorage.getItem('transactions'))
|| [];

function addTransaction(type, amount, category, description) {
  const transaction = {
    id: Date.now(),
    type: type, // 'income' ou 'expense'
    amount: parseFloat(amount),
    category: category,
    description: description,
    date: new Date().toISOString()
  };

  transactions.push(transaction);
  saveTransactions();
  updateUI();
}

function calculateBalance() {
  const income = transactions
    .filter(t => t.type === 'income')
    .reduce((sum, t) => sum + t.amount, 0);

  const expenses = transactions
    .filter(t => t.type === 'expense')
    .reduce((sum, t) => sum + t.amount, 0);

  return { income, expenses, balance: income - expenses };
```

```
}

function updateUI() {
  const { income, expenses, balance } = calculateBalance();

  document.getElementById('totalIncome').textContent =
    income.toFixed(2) + ' €';
  document.getElementById('totalExpenses').textContent =
    expenses.toFixed(2) + ' €';
  document.getElementById('balance').textContent =
    balance.toFixed(2) + ' €';

  displayTransactions();
  updateChart();
}

function updateChart() {
  // Utilisation de Chart.js
  const ctx =
document.getElementById('expenseChart').getContext('2d');

  // Grouper les dépenses par catégorie
  const categories = {};
  transactions
    .filter(t => t.type === 'expense')
    .forEach(t => {
      categories[t.category] = (categories[t.category] || 0) +
t.amount;
    });

  new Chart(ctx, {
    type: 'doughnut',
    data: {
      labels: Object.keys(categories),
      datasets: [{
        data: Object.values(categories),
        backgroundColor: ['#FF6384', '#36A2EB', '#FFCE56',
'#4BC0C0']
```

```
      }]
    }
  });
}
```

🔴 PROJET 10 — Portfolio Complet avec Dark Mode

📊 **Niveau : Avancé 🔴**

Technologies : HTML + CSS + JavaScript + LocalStorage • Durée : 8-10 heures • Difficulté : ⭐⭐⭐⭐⭐

Structure HTML (sections principales)

📄 **index.html**

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Mon Portfolio</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <!-- Navigation -->
  <nav class="navbar">
    <div class="logo">MonPortfolio</div>
    <ul class="nav-links">
      <li><a href="#home">Accueil</a></li>
      <li><a href="#about">À propos</a></li>
      <li><a href="#skills">Compétences</a></li>
      <li><a href="#projects">Projets</a></li>
      <li><a href="#contact">Contact</a></li>
    </ul>
    <button id="themeToggle">🌙</button>
    <button class="burger">≡</button>
  </nav>

  <!-- Hero Section -->
  <section id="home" class="hero">
    <h1 class="hero-title">Bonjour, je suis <span>Votre
```

```
Nom</span></h1>
    <p class="hero-subtitle">Développeur Web Full Stack</p>
    <a href="#contact" class="cta-btn">Me contacter</a>
  </section>

  <!-- À propos -->
  <section id="about" class="about">
    <h2>À propos de moi</h2>
    <p>Développeur passionné...</p>
  </section>

  <!-- Compétences -->
  <section id="skills" class="skills">
    <h2>Mes Compétences</h2>
    <div class="skill-item">
      <span>HTML/CSS</span>
      <div class="progress-bar">
        <div class="progress" style="width: 90%">90%</div>
      </div>
    </div>
    <!-- Répéter pour chaque compétence -->
  </section>

  <!-- Projets -->
  <section id="projects" class="projects">
    <h2>Mes Projets</h2>
    <div class="project-filters">
      <button data-filter="all">Tous</button>
      <button data-filter="web">Web</button>
      <button data-filter="mobile">Mobile</button>
    </div>
    <div class="projects-grid" id="projectsGrid"></div>
  </section>

  <!-- Contact -->
  <section id="contact" class="contact">
    <h2>Contactez-moi</h2>
    <form id="contactForm">
```

```
      <input type="text" placeholder="Nom" required>
      <input type="email" placeholder="Email" required>
      <textarea placeholder="Message" required></textarea>
      <button type="submit">Envoyer</button>
    </form>
  </section>

  <script src="script.js"></script>
</body>
</html>
```

Dark Mode (JavaScript)

📄 **script.js**

```
// Dark Mode Toggle
const themeToggle = document.getElementById('themeToggle');
const body = document.body;

// Charger le thème sauvegardé
const savedTheme = localStorage.getItem('theme');
if (savedTheme === 'dark') {
  body.classList.add('dark-mode');
  themeToggle.textContent = '☀';
}

themeToggle.addEventListener('click', () => {
  body.classList.toggle('dark-mode');

  if (body.classList.contains('dark-mode')) {
    localStorage.setItem('theme', 'dark');
    themeToggle.textContent = '☀';
  } else {
    localStorage.setItem('theme', 'light');
    themeToggle.textContent = '🌙';
  }
});
```

```
// Smooth Scroll
document.querySelectorAll('a[href^="#"]').forEach(anchor => {
  anchor.addEventListener('click', function(e) {
    e.preventDefault();
    const target =
document.querySelector(this.getAttribute('href'));
    target.scrollIntoView({ behavior: 'smooth' });
  });
});

// Menu mobile
const burger = document.querySelector('.burger');
const navLinks = document.querySelector('.nav-links');

burger.addEventListener('click', () => {
  navLinks.classList.toggle('active');
});

// Validation formulaire
const contactForm = document.getElementById('contactForm');
contactForm.addEventListener('submit', (e) => {
  e.preventDefault();
  alert('Message envoyé !');
  contactForm.reset();
});
```

🎉 FÉLICITATIONS !

Vous avez maintenant 10 projets complets pour votre portfolio !