RAPPORT DE GESTION DE PROJET

–

CINE TECH

Réalisé par :

- Rania Bakkach
- Doaa Maskaoui
- Aya Moujahid

Encadré par :

- Mme Khadija Tlemçani

Résumé

Ce rapport présente la gestion du projet **CineTech**, une application web de gestion de films et de réalisateurs. L'objectif principal est de développer une application côté client offrant une interface moderne, des statistiques interactives, et une intégration avec une API externe (OMDB). Le projet illustre la mise en pratique des connaissances en HTML, CSS et JavaScript, la manipulation dynamique du DOM et la persistance des données via le **localStorage**.

---

1. Introduction

Présentation générale :
CineTech est un **Dashboard interactif** pour la gestion complète d'un catalogue de films et de réalisateurs. L'application permet :

- La gestion complète des films (CRUD : créer, lire, mettre à jour, supprimer)
- La gestion des réalisateurs et de leurs informations
- La visualisation de statistiques via des **graphiques dynamiques**
- L'importation de films depuis une **API (OMDB)**

---

2. Objectifs du rapport

Objectif	Description
Développement front-end	Appliquer les concepts du HTML, CSS et JavaScript dans un projet réel
Manipulation du DOM	Modifier dynamiquement le contenu et gérer les interactions utilisateur
Persistance des données	Stocker et récupérer les informations directement dans le navigateur
API externe	Consommer l'API OMDB pour récupérer des informations sur les films
Visualisation	Présenter des statistiques interactives via des graphiques
Interface moderne	Créer un Dashboard responsive et ergonomique

---

3. Technologies

HTML → Structure et organisation des pages

CSS → Mise en forme et design responsive

JavaScript→ Logique applicative et manipulation du DOM

Chart.js→ Création de graphiques statistiques

Font Awesome → Icônes pour boutons et menus

OMDB API → Récupération des informations de films depuis Internet

localStorage → Sauvegarde et persistance des données côté client

---

4. Architecture du projet

- **index.html** → Structure principale de l'application
- **style.css** → Mise en forme et animations
- **app.js** → Logique JavaScript pour le CRUD, Dashboard et API

---

5. Fonctionnalités principales

1. Dashboard

- Affiche les **indicateurs clés (KPI)** :
  - Nombre total de films
  - Note moyenne
  - Nombre de réalisateurs
  - Nombre de films récents (≥ 2024)
- Graphiques dynamiques générés avec **Chart.js** :
  - Répartition des films par genre (Bar Chart)
  - Évolution des notes des films (Line Chart)

2. Gestion des films (CRUD)

- Ajouter, modifier, supprimer et afficher les films
- Chaque film contient :
  - Titre, année, durée, genre, note, réalisateur, synopsis, affiche
- Fonctionnalités avancées :
  - Recherche en temps réel
  - Tri par titre, année ou note
  - Filtrage par genre

3. Gestion des réalisateurs

- Ajouter, modifier et supprimer les réalisateurs
- Les réalisateurs liés à des films ne peuvent pas être supprimés
- Informations : nom, nationalité, biographie

4. Intégration API OMDB

- Rechercher des films par titre via l'API OMDB
- Affichage des résultats sous forme de cartes
- Possibilité d'importer un film dans l'application

---

6. Points forts du projet

- **Interface moderne et ergonomique** : Sidebar fixe, animations CSS, thème sombre
- **Architecture claire** : Séparation HTML / CSS / JS et fonctions organisées
- **Données persistantes** : Utilisation de localStorage pour conserver les informations
- **Interactivité** : Modals dynamiques, mise à jour automatique du Dashboard
- **API externe** : Recherche et importation de films via OMDB

---

7. Fonctions principales du code

1. Gestion des films

- `openMovieModal()` → ouvre le modal d'ajout/modification
- `saveMovie()` → enregistre un film dans le localStorage
- `deleteMovie()` → supprime un film après confirmation
- `displayMovies()` → affiche les films sous forme de cartes
- `showMovieDetail()` → affiche les détails d'un film
- `validateMovieForm()` → vérifie les champs du formulaire

2. Gestion des réalisateurs

- `openDirectorModal()` → ouvre le modal pour les réalisateurs
- `saveDirector()` → ajoute/modifie un réalisateur
- `deleteDirector()` → supprime un réalisateur si possible
- `displayDirectors()` → affiche les réalisateurs dans le tableau
- `loadDirectorOptions()` → charge les réalisateurs dans le menu déroulant

3. Dashboard & statistiques

- `updateDashboard()` → met à jour les KPI
- `updateGenreChart()` → met à jour le graphique par genre
- `updateRatingChart()` → met à jour le graphique des notes

4. API externe

- `searchOMDB()` → recherche de films via OMDB
- `importMovie()` → importation d'un film dans l'application

5. Utilitaires

- `showError()` → affiche un message d'erreur
- `clearErrors()` → supprime les messages d'erreurs

---

8. Structure HTML et CSS

2.1 Navbar et Sidebar

Le design repose sur une **navbar** fixe en haut et une **sidebar** latérale, permettant la navigation entre les sections Dashboard, Films, Réalisateurs et API OMDB.

```css
.navbar {
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    height: 60px;
    background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 0 30px;
    z-index: 1000;
    box-shadow: 0 4px 20px rgba(102, 126, 234, 0.3);
}
```

```css
.sidebar {
    position: fixed;
    top: 0;
    left: 0;
    width: 260px;
    height: 100vh;
    background: #1a1a2e;
    padding: 20px 0;
    border-right: 1px solid #2a2a3e;
    overflow-y: auto;
}
```

```css
.sidebar .menu li.active,
.sidebar .menu li:hover {
    background: #2a2a3e;
    padding-left: 30px;
    border-left: 4px solid #667eea;
}
```

Figure 1 – Extrait CSS pour la navigation et la sidebar

2.2 Main Content et Sections

Chaque section (Dashboard, Films, Réalisateurs) est masquée par défaut et affichée via la classe `active` :

```css
.section {
    display: none;
    animation: fadeIn 0.4s;
}

.section.active {
    display: block;
}

.section-title {
    font-size: 32px;
    margin-bottom: 30px;
    color: #fff;
    display: flex;
    align-items: center;
    gap: 15px;
}
```

Figure 2 – Structure CSS des sections principales

2.3 KPI Cards et Tables

Le dashboard présente des **KPI Cards** et des tableaux stylisés :

```css
.kpi-card {
    background: linear-gradient(135deg, #1a1a2e 0%, #2a2a3e 100%);
    padding: 25px;
    border-radius: 15px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    box-shadow: 0 8px 20px rgba(0, 0, 0, 0.3);
    border: 1px solid #2a2a3e;
    transition: transform 0.3s, box-shadow 0.3s;
}
```

```css
.table th,
.table td {
    padding: 12px;
    border-bottom: 1px solid #2a2a3e;
    color: #ccc;
    text-align: left;
}
```

```css
.table tbody tr:hover {
    background: #2a2a3e;
}
```

Figure 3 – Extrait CSS pour KPI et tableaux

---

3. Gestion des Films

3.1 Modal Film

Le formulaire d'ajout ou de modification de film est présenté via un modal :

```html
<div class="modal" id="movieModal">
    <div class="modal-content">
        <h2 id="movieModalTitle">Ajouter un Film</h2>
        <form id="movieForm" onsubmit="event.preventDefault(); saveMovie();">
            <input type="hidden" id="movieId">
            <label>Titre <span class="required">*</span></label>
            <input type="text" id="movieTitle" class="form-control">
            <div id="movieTitleError" class="error-message"></div>

            <label>Année <span class="required">*</span></label>
            <input type="number" id="movieYear" class="form-control">
            <div id="movieYearError" class="error-message"></div>

            <label>Durée (min) <span class="required">*</span></label>
            <input type="number" id="movieDuration" class="form-control">
            <div id="movieDurationError" class="error-message"></div>

            <label>Genre <span class="required">*</span></label>
            <select id="movieGenre" class="form-control">
                <option value="">Sélectionner un genre</option>
                <option value="Action">Action</option>
                <option value="Comédie">Comédie</option>
                <option value="Drame">Drame</option>
                <option value="Science-Fiction">Science-Fiction</option>
                <option value="Horreur">Horreur</option>
                <option value="Thriller">Thriller</option>
                <option value="Animation">Animation</option>
            </select>
            <div id="movieGenreError" class="error-message"></div>

            <label>Note (0-10) <span class="required">*</span></label>
            <input type="number" step="0.1" id="movieRating" class="form-control">
            <div id="movieRatingError" class="error-message"></div>

            <label>Réalisateur <span class="required">*</span></label>
            <select id="movieDirector" class="form-control"></select>
            <div id="movieDirectorError" class="error-message"></div>

            <label>Synopsis</label>
            <textarea id="movieSynopsis" class="form-control"></textarea>

            <label>Poster (URL)</label>
            <input type="text" id="moviePoster" class="form-control">

            <div class="modal-actions">
                <button type="button" class="btn btn-secondary" onclick="closeMovieModal()">Annuler</button>
                <button type="submit" class="btn btn-primary">Enregistrer</button>
            </div>
        </form>
    </div>
</div>
```

Figure 4 – Modal HTML pour ajouter/modifier un film

3.2 Validation du Formulaire

```javascript
function validateMovieForm() {
    let isValid = true;
    clearErrors();

    const title = document.getElementById('movieTitle').value.trim();
    const year = parseInt(document.getElementById('movieYear').value);
    const duration = parseInt(document.getElementById('movieDuration').value);
    const genre = document.getElementById('movieGenre').value;
    const rating = parseFloat(document.getElementById('movieRating').value);
    const director = document.getElementById('movieDirector').value;

    if (!title) { showError('movieTitle', 'Le titre est requis'); isValid = false; }
    if (!year || year < 1900 || year > 2026) { showError('movieYear', 'Année invalide'); isValid = false; }
    if (!duration || duration < 1) { showError('movieDuration', 'Durée invalide'); isValid = false; }
    if (!genre) { showError('movieGenre', 'Le genre est requis'); isValid = false; }
    if (!rating || rating < 0 || rating > 10) { showError('movieRating', 'Note invalide (0-10)'); isValid = false; }
    if (!director) { showError('movieDirector', 'Le réalisateur est requis'); isValid = false; }

    return isValid;
}
```

Figure 5 – Validation des champs du formulaire film

3.3 Affichage des Films

```
function displayMovies() {
    const searchTerm = document.getElementById('searchMovie').value.toLowerCase();
    const sortValue = document.getElementById('sortMovie').value;
    const filterGenre = document.getElementById('filterGenre').value;

    let filtered = movies.filter(m =>
        (m.title.toLowerCase().includes(searchTerm) || (m.synopsis?.toLowerCase().includes(searchTerm))) &&
        (!filterGenre || m.genre === filterGenre)
    );

    // Sorting
    const sortMap = {
        'title-asc': (a,b)=> a.title.localeCompare(b.title),
        'title-desc': (a,b)=> b.title.localeCompare(a.title),
        'year-asc': (a,b)=> a.year-b.year,
        'year-desc': (a,b)=> b.year-a.year,
        'rating-asc': (a,b)=> a.rating-b.rating,
        'rating-desc': (a,b)=> b.rating-a.rating
    };
    if (sortMap[sortValue]) filtered.sort(sortMap[sortValue]);

    const container = document.getElementById('moviesList');
    if (!filtered.length) {
        container.innerHTML = '<p style="text-align:center; padding:40px; color:#888;">Aucun film trouvé</p>';
        return;
    }

    container.innerHTML = filtered.map(movie => `
        <div class="movie-card" onclick="showMovieDetail(${movie.id})">
            <img src="${movie.poster}" class="movie-poster" alt="${movie.title}">
            <div class="movie-body">
                <div class="movie-title">${movie.title}</div>
                <div class="movie-year">${movie.year} • ${movie.genre}</div>
                <div class="movie-rating">⭐ ${movie.rating}/10</div>
                <div class="movie-actions">
                    <button class="btn btn-sm btn-warning" onclick="event.stopPropagation(); openMovieModal(${movie.i
                    <button class="btn btn-sm btn-danger" onclick="event.stopPropagation(); deleteMovie(${movie.id})"
                </div>
            </div>
        </div>
```

Figure 6 – Fonction d'affichage des films dans une grille

4. Gestion des Réalisateurs

4.1 Modal Réalisateur

```html
<div class="modal" id="directorModal">
    <div class="modal-content">
        <h2 id="directorModalTitle">Ajouter un Réalisateur</h2>
        <form id="directorForm" onsubmit="event.preventDefault(); saveDirector();">
            <input type="hidden" id="directorId">
            <label>Nom <span class="required">*</span></label>
            <input type="text" id="directorName" class="form-control">

            <label>Nationalité <span class="required">*</span></label>
            <input type="text" id="directorNationality" class="form-control">

            <label>Bio</label>
            <textarea id="directorBio" class="form-control"></textarea>

            <div class="modal-actions">
                <button type="button" class="btn btn-secondary" onclick="closeDirectorModal()">Annuler</button>
                <button type="submit" class="btn btn-primary">Enregistrer</button>
            </div>
        </form>
    </div>
</div>
```

Figure 7 – Modal HTML pour ajouter/modifier un réalisateur

4.2 Affichage des Réalisateurs

```
function displayDirectors() {
    const tbody = document.getElementById('directorsList');
    if (!directors.length) {
        tbody.innerHTML = '<tr><td colspan="4" style="text-align:center">Aucun réalisateur</td></tr>';
        return;
    }

    tbody.innerHTML = directors.map(d => {
        const movieCount = movies.filter(m => m.director == d.id).length;
        return `
            <tr>
                <td style="color:#e50914"><strong>${d.name}</strong></td>
                <td>${d.nationality}</td>
                <td>${movieCount}</td>
                <td>
                    <button class="btn btn-sm btn-warning" onclick="openDirectorModal(${d.id})">✏️</button>
                    <button class="btn btn-sm btn-danger" onclick="deleteDirector(${d.id})">🗑️</button>
                </td>
            </tr>
        `;
    }).join('');
}
```

Figure 8 – Fonction d'affichage des réalisateurs dans un tableau

---

4.3 Gestion des options des réalisateurs

```
function loadDirectorOptions() {
    const select = document.getElementById('movieDirector');
    select.innerHTML = '<option value="">Sélectionner un réalisateur</option>' +
        directors.map(d => `<option value="${d.id}">${d.name}</option>`).join('');
}
```

```
function findOrCreateDirector(directorName) {
    const director = directors.find(d => d.name.toLowerCase() === directorName.toLowerCase());
    if (director) return director.id;

    // Create new director
    const newDirector = {
        id: Date.now(),
        name: directorName,
        nationality: 'Inconnue',
        bio: ''
    };
    directors.push(newDirector);
    localStorage.setItem(STORAGE_KEYS.DIRECTORS, JSON.stringify(directors));
    return newDirector.id;
}
```

Figure 9 – Fonctions utilitaires pour les réalisateurs

---

9. Conclusion

L’application **CineTech** permet une gestion complète des films et réalisateurs avec une interface intuitive et moderne. L’utilisation de **localStorage** assure la persistance des données, et les fonctionnalités de recherche et d’importation via l’API OMDB enrichissent le catalogue.

Les modals, formulaires et validations garantissent la cohérence des données saisies, tandis que les **KPI et graphiques**fournissent des statistiques visuelles claires pour le suivi des films et des réalisateurs.

Remerciements

Nous exprimons notre profonde gratitude à Mme Khadija Tlemçani pour son encadrement attentif, ses conseils éclairés et sa disponibilité constante tout au long de la réalisation de ce projet. Son soutien précieux a largement contribué à la réussite de notre travail ainsi qu’au perfectionnement de nos compétences en développement web.