AEC

Développement Web

Projet de session

Automne – 2025

Programme	LEA.6C Développement Web

Titre du cours	***Modification d'un système existant***
Numéro du cours	***420-7A3-FE***
Numéro de cohorte	**07019**
Session	*Automne 2025*
Pondération	*10% , 10% , 10 % , 30%*
Date de soumission	*15-12-2025*
Dates de présentation	*16-12-2025*

Projet	CinéManage -plateforme Web pour la gestion des films

Introduction générale du projet

1. Objectif du projet

Le projet consiste à analyser et améliorer un système existant, en refondant certains modules et en développant de nouveaux, afin de le rendre plus fonctionnel, ergonomique et sécurisé.
L'objectif est de moderniser l'architecture, optimiser la base de données, renforcer la sécurité et améliorer l'expérience utilisateur sur les interfaces publiques et administratives.
Ce projet permettra aux étudiants de mettre en pratique leurs compétences en développement web**,** programmation côté client et serveur**,** gestion de bases de données**, et** tests fonctionnels et unitaires, tout en respectant les bonnes pratiques professionnelles et les exigences du demandeur de service.

2. Matériel existant

Avant d'entamer la refonte, une analyse du système actuel est réalisée à partir des ressources disponibles afin d'identifier ses forces**,** faiblesses **et** axes d'amélioration.

Éléments disponibles :

- **Code source existant :** Comprend la structure actuelle du projet (fichiers backend, frontend, scripts, etc.) et permet d'analyser l'architecture logicielle et la logique des modules existants. (Télécharger ici)
- **Documentation métier :** Présente les besoins fonctionnels, les processus métiers, les profils utilisateurs et les objectifs du système. Elle servira de base pour identifier les fonctionnalités à conserver, à corriger ou à refondre.(Consulter ici)
- **Documentation technique :** Décrit la configuration du système, les modules en place, la structure de la base de données et les technologies employées. (Consulter ici)

3. Construction des groupes

Les étudiants sont répartis en groupes de 3 à 5 membres pour favoriser la collaboration et la répartition des responsabilités selon les compétences de chacun :

- **Développement front-end** : mise à jour et amélioration des interfaces existantes, intégration de nouvelles fonctionnalités, responsive design.
- **Développement back-end** : refonte des modules existants, création de nouveaux modules, gestion de la logique applicative et de la base de données.

- **Gestion de projet et documentation** : planification des sprints, suivi des tâches via Trello, rédaction de la documentation technique et utilisateur.

La collaboration se fait via GitHub pour le versioning et les merges de code, et via Trello pour le suivi des tâches et l'organisation des sprints.

4. Méthodologie de travail

Le projet est organisé en sprints itératifs, avec des livrables précis pour chaque évaluation. La méthodologie adoptée est inspirée des pratiques **Agile**, permettant :

- Une analyse et planification progressive des modifications à apporter au système existant.
- Des revues régulières du code et des fonctionnalités modifiées.
- Une intégration continue des modules développés et refondus.
- Des tests unitaires et fonctionnels pour chaque module avant livraison.

Chaque sprint correspond à une évaluation et est conçu pour être fonctionnel et testé**,** garantissant que le système évolue de manière cohérente vers une version finale améliorée.

5. Évaluations

Évaluation 1 : Analyse et préparation du projet ***Mardi 4 novembre***

Objectifs :

- Mener une analyse approfondie des besoins et du système existant.
- Identifier les modules à refondre ainsi que les nouveaux modules à concevoir.
- Analyser les limites et points faibles du système actuel, notamment :
    - les user flows (parcours utilisateurs associés à chaque user story) ;
    - les interfaces graphiques ;
    - la structure et la conception de la base de données ;
    - **l'**architecture du code ;
    - la documentation technique et fonctionnelle.
- Reformuler les limites identifiées sous forme de besoins (backlog, user stories).
- Préparer l'environnement de développement : installation du serveur local, configuration de la base de données et de l'IDE.
- Créer et configurer le repository GitHub, en définissant le workflow collaboratif (structure des branches, conventions de commits, etc.).

- Mettre en place un outil de planification (Trello) pour organiser le suivi des tâches et la répartition du travail.

Livrables attendus :

- Cahier des besoins révisé**,** intégrant l'analyse complète du système existant.
- Schéma initial de l'architecture MVC *(facultatif)*.
- Repository GitHub configuré et tableau Trello opérationnel.

Modalités de soumission :

- Un document PDF comprenant :
  - le cahier des besoins et **l'**analyse du système existant (diagrammes, user stories) ;
  - le schéma MVC *(si applicable)*.
- Le lien du repository GitHub et une capture d'écran du tableau Trello.
- Une présentation orale des conclusions de l'analyse et du plan de travail pour la suite du projet.

Évaluation 2 : Conception et développement du projet ***Mardi 18 novembre***

Partie 1 – Conception des solutions et maquettes

Objectifs :

- Élaborer la base des pages front-end**,** incluant **l'**interface publique et **l'**interface d'administration**.**
- Créer les premières maquettes UI/UX, avec une attention particulière portée à la navigation et à **l'**ergonomie.
- Prioriser les modules identifiés dans le backlog en vue des prochains sprints.
- Concevoir la structure MVC complète (Model, View, Controller).

Partie 2 – Structure de code et modules de base (Sprint 1)

Objectifs :

- Implémenter le squelette MVC du projet.
- Effectuer un refactoring complet du module "Films" (CRUD), en intégrant la sécurité et les validations nécessaires.
- Développer le module d'authentification sécurisée avec gestion des rôles utilisateurs**.**

- Mettre en place une interface d'administration fonctionnelle permettant la gestion des films**.**

Livrables attendus :

- Documentation fonctionnelle incluant :
  - les maquettes des interfaces publique et d'administration ;
  - le schéma de l'architecture MVC ;
  - le backlog priorisé.
- Code source MVC structuré et organisé**,** couvrant :
  - le module de gestion des films ;
  - le module d'authentification.
- Documentation technique pour les modules développés (structure, sécurité, logique métier, etc.).

Modalités de soumission :

- Une documentation métier et technique complète.
- Une documentation technique détaillée pour les modules développés.
- Le code source complet (modules *Films* et *Authentification*) disponible sur GitHub.
- Une présentation orale portant sur les choix UX/UI et les décisions architecturales.

Évaluation 3 : Développement des nouveaux modules et intégration de la base de données

Mardi 2 décembre

Partie 1 – Développement des modules et base de données (Sprint 2)

Objectifs :

- Créer ou modifier la base de données afin d'intégrer les nouvelles tables nécessaires.
- Développer un module fonctionnel conformément au backlog défini.
- Connecter le backend à la base de données et valider les requêtes SQL à travers des tests de cohérence et d'intégrité des données.

Partie 2 – Développement du 2eme module et interface publique (Sprint 3)

Objectifs :

- Développer un nouveau module selon les priorités du backlog.
- Améliorer l'interface publique du site, notamment :
  - la consultation des films ;
  - le filtrage et la recherche ;
  - la mise en page responsive.
- Intégrer les médias (affiches, bandes-annonces, visuels).
- Réaliser les tests unitaires et d'intégration sur l'ensemble des modules développés.

Livrables attendus :

- Base de données complète et normalisée**.**
- Modules pleinement fonctionnels côté administration et interface publique.
- Interface utilisateur moderne, fluide et responsive.
- Documentation technique détaillant :
  - les schémas de la base de données ;
  - les requêtes SQL principales ;
  - les modules et leur logique de fonctionnement**.**
  - Tests validés pour tous les modules développés.

Modalités de soumission :

- Code source complet des modules et de l'interface publique.
- Documentation technique finale (PDF).
- Présentation orale comprenant :
  - une démonstration fonctionnelle de la partie administration ;
  - une démonstration interactive de l'interface publique.

Évaluation 4 (Finale) : Livraison finale et documentation

Mardi 16 décembre

Objectifs :

- Effectuer une révision complète du projet et apporter les corrections finales, incluant les tests de validation.
- Assurer **l**'intégration complète de tous les modules au sein d'un système cohérent et stable.
- Rédiger l'ensemble de la documentation technique ainsi qu'un guide utilisateur clair et détaillé (administration et interface publique).
- Préparer la présentation finale et la démonstration complète des fonctionnalités du projet.

Livrables attendus :

- Application complète et fonctionnelle, incluant le backend et le frontend.
- Documentation technique complète, comprenant :
    - le schéma MVC final,
    - le schéma de la base de données,
    - la description de l'API interne,
    - la documentation des tests (unitaires et d'intégration).
- Documentation utilisateur destinée :
    - aux administrateurs,
    - aux utilisateurs publics.
- Démonstration finale prête pour l'évaluation.

Modalités de soumission :

- Code source complet disponible sur GitHub.
- Documentation technique (PDF)**.**
- Documentation utilisateur (PDF)**.**
- Présentation et démonstration finale en classe.

Notes supplémentaires

- Chaque sprint doit produire un **module fonctionnel et testé**.
- Collaboration via GitHub avec branches pour chaque module et merge requests.

- Suivi via Trello avec statut “À faire / En cours / Terminé” pour chaque tâche.
- Les livrables et soumissions sont exigés avant l’évaluation correspondante.

a. Tableau récapitulatif de la pondération des évaluations

Évaluation	Date	Date de soumission	Pondération
Évaluation 1	Mardi 4 novembre	Mardi 4 novembre	10%
Évaluation 2	Mardi 18 novembre	Lundi 17 novembre	10%
Évaluation 3	Mardi 2 décembre	Lundi 1er décembre	10%
Évaluation 4 (Finale)	Mardi 16 décembre	Lundi 15 décembre	30%

Remarque : Les évaluations intermédiaires permettent de valider l’avancement du projet et les compétences acquises sur la refonte et l’amélioration du système existant, tandis que l’évaluation finale porte sur l’intégration complète, la qualité de l’application et la documentation.

I. Description métier du contexte du système existant

1. Présentation générale

CinéManage est une plateforme Web conçue pour la gestion des films au sein d'un cinéma. Elle permet à la fois l'administration interne et la consultation publique du catalogue de films. L'objectif principal du système est de centraliser les informations relatives aux films diffusés et de simplifier leur gestion, tout en offrant au public un accès facile et rapide aux contenus cinématographiques proposés.

Sur le plan fonctionnel, le système actuel offre une gestion basique des films : les administrateurs peuvent créer, lire, mettre à jour et supprimer des fiches de films (CRUD). Les utilisateurs publics peuvent consulter la liste des films disponibles et visualiser certaines informations associées, telles que le titre, la description et l'année de sortie. Cependant, des fonctionnalités essentielles comme la réservation de places, la consultation des horaires de projection ou l'achat de billets en ligne ne sont pas encore prises en charge. La plateforme inclut un mécanisme d'authentification rudimentaire permettant aux administrateurs de se connecter et de gérer le contenu, mais elle ne dispose d'aucun module pour la gestion des salles ou des tickets. Cette absence de fonctionnalités complètes limite la planification interne et empêche une coordination efficace des séances et des ventes.

L'interface utilisateur présente plusieurs lacunes importantes. Elle ne respecte pas les normes actuelles en matière d'ergonomie et de conception UX/UI. La navigation est peu intuitive, la disposition des éléments manque de cohérence et la structure de l'information ne favorise pas une consultation fluide. De plus, l'affichage n'est pas responsive, ce qui complique l'utilisation du site sur les appareils mobiles et nuit à l'expérience globale des utilisateurs. L'absence d'un design adaptatif et d'une hiérarchisation claire de l'information rend la plateforme difficile à exploiter tant pour les administrateurs que pour le public.

Sur le plan technique, CinéManage repose sur une architecture monolithique développée en **PHP procédural**, utilisant une base de données **MySQL** et des scripts **JavaScript** intégrés directement dans les pages HTML. L'absence de séparation claire entre la logique métier, la présentation et la gestion des données rend la maintenance complexe et l'évolution du système laborieuse. Aucune structure de type **MVC (Modèle–Vue–Contrôleur)** n'a été mise en place, ce qui limite la modularité et la réutilisabilité du code. Par ailleurs, le projet ne dispose pas de système de gestion de versions (Git), ce qui complique le travail collaboratif et le suivi des modifications. Enfin, la documentation technique est très limitée, ce qui rend difficile la compréhension de la structure du projet et l'intégration de nouvelles fonctionnalités.

En résumé, le système CinéManage constitue une base fonctionnelle pour la gestion de films, mais ses limites techniques et ergonomiques freinent son exploitation et son évolution. L'absence de fonctionnalités telles que la gestion des salles, la programmation des séances ou la billetterie en ligne réduit considérablement sa valeur opérationnelle. Une refonte complète du système est donc nécessaire afin de moderniser son architecture, d'améliorer l'expérience utilisateur et d'intégrer les pratiques modernes de développement, tout en assurant la pérennité et la sécurité de la plateforme.

2. Stories et Backlog du système existant

2.1. User Stories du système existant

Le système CinéManage repose actuellement sur un ensemble limité de fonctionnalités centrées sur la gestion des films et l'accès public à leur catalogue. Les **user stories suivantes** illustrent les besoins pris en charge dans la version actuelle du système, ainsi que leurs limites.

Rôle : Administrateur

User Story 1 — Authentification de l'administrateur

En tant qu'administrateur, je veux pouvoir me connecter au système à l'aide de mes identifiants afin d'accéder à l'interface d'administration.

- **Critères d'acceptation :**
    - L'administrateur saisit un nom d'utilisateur et un mot de passe.
    - Le système vérifie les informations d'identification dans la base de données.
    - L'accès est accordé si les informations sont valides ; sinon, un message d'erreur s'affiche.

User Story 2 — Ajout d'un film

En tant qu'administrateur, je veux pouvoir ajouter un nouveau film avec ses informations principales afin qu'il apparaisse dans la liste publique.

- **Critères d'acceptation :**
    - Le formulaire permet de saisir : titre, réalisateur, genre, année de sortie et description.
    - Le film est ajouté à la base de données et visible dans la liste publique.

User Story 3 — Modification d'un film

En tant qu'administrateur, je veux pouvoir modifier les informations d'un film existant afin de corriger ou mettre à jour son contenu.

- **Critères d'acceptation :**
    - L'administrateur sélectionne un film depuis la liste.
    - Les informations existantes s'affichent dans un formulaire modifiable.
    - Les modifications sont enregistrées dans la base de données.

User Story 4 — Suppression d'un film

En tant qu'administrateur, je veux pouvoir supprimer un film du catalogue afin de maintenir la base de données à jour.

- **Critères d'acceptation :**
    - L'administrateur clique sur un bouton "Supprimer".
    - Le film est supprimé définitivement de la base de données.

User Story 5 — Consultation de la liste des films (interface d'administration)

En tant qu'administrateur, je veux pouvoir consulter la liste complète des films afin de gérer efficacement le catalogue.

- **Critères d'acceptation :**
    - Les films sont affichés sous forme de tableau.
    - Chaque ligne contient les informations de base (titre, réalisateur, année).
    - Des boutons permettent de modifier ou supprimer un film.

Rôle : Utilisateur public

User Story 6 — Consultation du catalogue de films

En tant qu'utilisateur, je veux pouvoir consulter la liste des films disponibles afin de connaître les œuvres actuellement diffusées au cinéma.

- **Critères d'acceptation :**
    - Les films sont affichés sur la page publique.
    - Chaque film affiche son titre, son réalisateur, et une brève description.

User Story 7 — Consultation du détail d'un film

En tant qu'utilisateur, je veux pouvoir consulter les détails d'un film afin d'obtenir plus d'informations sur celui-ci.

- **Critères d'acceptation :**
    - Un clic sur un film affiche sa fiche détaillée.
    - La fiche contient des informations textuelles de base (titre, réalisateur, année, description).

2.2. Backlog du système existant

Le backlog suivant recense les **fonctionnalités actuellement présentes dans le système** et celles **qui manquent ou nécessitent une amélioration**.

Catégorie	Fonctionnalités existantes
Authentification	Connexion basique pour les administrateurs
Gestion des films	CRUD complet (ajout, lecture, modification, suppression)
Interface publique	Liste des films consultable par le public
Interface d'administration	Liste des films avec actions de gestion
Gestion des salles	Aucune

Catégorie	Fonctionnalités existantes
Programmation des séances	Aucune
Billetterie / Tickets	Aucune
Base de données	Tables films et administrateurs
Architecture du code	PHP procédural, HTML, CSS, JS intégré
Contrôle de versions	Aucun
Documentation	Presque inexistante

Le système CinéManage, dans son état actuel, répond uniquement à un besoin minimal de gestion et de consultation de films. Il ne permet ni la planification des projections, ni la gestion des salles, ni la vente de billets.

3. User Flows du système existant CinéManage

Le système **CinéManage** permet deux types d'utilisateurs :

- **L'administrateur**, qui gère les films à travers un espace d'administration.
- **L'utilisateur public**, qui consulte la liste des films disponibles.

Chaque profil suit un parcours fonctionnel distinct détaillé ci-dessous.

3.1. Flux — Authentification de l'administrateur

Permettre à un administrateur autorisé d'accéder à l'espace d'administration.

Description du flux :

1. L'administrateur accède à la page de connexion via un lien dédié.
2. Il saisit son **nom d'utilisateur** et son **mot de passe**.
3. Le système envoie une requête vers la base de données pour vérifier la correspondance des identifiants.
4.
    - Si les informations sont **valides**, l'administrateur est redirigé vers le tableau de bord d'administration.
    - Si elles sont **invalides**, un message d'erreur s'affiche ("Identifiants incorrects").

3.2. Flux — Ajout d'un film (CRUD : Create)

Permettre à l'administrateur d'ajouter un nouveau film dans le catalogue.

Description du flux :

1. L'administrateur, depuis son tableau de bord, clique sur "Ajouter un film".
2. Un **formulaire** s'affiche avec les champs :
    - Titre
    - Réalisateur
    - Genre
    - Année de sortie
    - Description
3. Il remplit les champs et soumet le formulaire.
4. Le système enregistre les informations dans la base de données.
5. Le nouveau film apparaît instantanément dans la liste des films administratifs et dans le catalogue public.

3.3. Flux — Modification d'un film (CRUD : Update)

Permettre à l'administrateur de mettre à jour les informations d'un film existant.

Description du flux :

1. L'administrateur accède à la liste des films.
2. Il sélectionne le film à modifier et clique sur "Modifier".
3. Les informations actuelles du film s'affichent dans un formulaire.
4. Il effectue les changements nécessaires puis soumet le formulaire.
5. Le système met à jour les informations dans la base de données et recharge la liste des films.

4. Flux — Suppression d'un film (CRUD : Delete)

Permettre à l'administrateur de supprimer un film du catalogue.

Description du flux :

1. L'administrateur consulte la liste des films.
2. Il clique sur le bouton "Supprimer" associé au film à retirer.

3. Le système exécute une requête SQL de suppression dans la base de données.
4. Le film disparaît immédiatement de la liste publique et de l'espace d'administration.

5. Flux — Consultation du catalogue (Administrateur)

Afficher la liste complète des films enregistrés pour permettre la gestion.

Description du flux :

1. Une fois connecté, l'administrateur accède à la page "Liste des films".
2. Le système récupère toutes les entrées de la table films.
3. Les données sont affichées dans un tableau listant : titre, réalisateur, année.
4. Des actions "Modifier" et "Supprimer" sont disponibles pour chaque ligne.

6. Flux — Consultation du catalogue de films (Utilisateur public)

Permettre au public de consulter les films disponibles sans connexion.

Description du flux :

1. L'utilisateur accède à la page d'accueil publique du site.
2. Le système interroge la base de données pour récupérer la liste des films.
3. Les films sont affichés sous forme de liste ou de grille avec :
    - Titre
    - Réalisateur
    - Brève description
4. L'utilisateur peut cliquer sur un film pour afficher sa fiche détaillée.

7. Flux — Consultation du détail d'un film (Utilisateur public)

Afficher des informations détaillées sur un film sélectionné.

Description du flux :

1. L'utilisateur clique sur un film depuis la liste publique.
2. Le système charge la page de détails en récupérant les informations du film via son identifiant.
3. La fiche du film affiche : titre, réalisateur, année, genre, description.

II. Documentation technique du système existant — CinéManage

1. Présentation générale

Le système **CinéManage** est une application Web interne et publique conçue pour la gestion et la consultation des films d'un cinéma.

Son objectif principal est de permettre aux administrateurs de gérer les informations des films (ajout, modification, suppression) et au public de consulter la liste des films disponibles.

Le système a été développé en **PHP procédural** avec une **base de données MySQL**, sans utilisation d'architecture MVC ni de framework.

Bien qu'il soit fonctionnel, il présente plusieurs limitations techniques liées à l'absence de modularité, de gestion de versions, et de bonnes pratiques de développement modernes.

2. Architecture générale du système

Le système repose sur une **architecture monolithique** dans laquelle la logique applicative, la présentation et la gestion des données sont fortement couplées. Chaque page PHP contient à la fois le code HTML, la logique de traitement et les requêtes SQL.

Structure de l'application :

- **Frontend** : pages HTML/CSS statiques intégrées directement dans les fichiers PHP.
- **Backend** : logique métier et traitements en PHP procédural (validation, CRUD, authentification).
- **Base de données** : MySQL locale, sans contrainte d'intégrité ni relations entre tables.

Fonctionnement général :

1. L'utilisateur (public ou administrateur) accède à une page via le navigateur.
2. Le serveur Apache exécute le script PHP correspondant.
3. Le script se connecte directement à la base MySQL via mysqli_connect().
4. Les résultats sont affichés dynamiquement à travers du code HTML.

3. Technologies utilisées

Composant	Technologie	Version / Détail
Langage serveur	PHP procédural	v7.3
Serveur Web	Apache	v2.4
Base de données	MySQL	v5.7
Frontend	HTML5 / CSS3 / JavaScript natif	—
Hébergement	Serveur local (XAMPP / WAMP)	—
Navigateur recommandé	Chrome / Firefox	dernières versions

Aucun framework ni gestionnaire de dépendances (Composer, npm, etc.) n'est utilisé. Le code est directement déployé sur le serveur de production via FTP.

4. Structure du code source

```
/cinemanage/
│
├── index.php              # Page d'accueil (liste publique des films)
├── film.php               # Page de détails d'un film
│
├── admin/
│   ├── login.php          # Page de connexion administrateur
│   ├── dashboard.php      # Interface principale de gestion
│   ├── add_film.php       # Ajout d'un film
│   ├── edit_film.php      # Modification d'un film
│   ├── delete_film.php    # Suppression d'un film
│   └── logout.php         # Déconnexion
│
├── includes/
│   ├── db_connect.php     # Script de connexion MySQL
│   ├── header.php         # En-tête HTML commun
│   └── footer.php         # Pied de page commun
│
├── assets/
│   ├── css/style.css      # Feuille de style
│   └── js/script.js       # Scripts JavaScript
│
└── config.php             # Paramètres de configuration (base de données)
```

5. Base de données

Nom de la base : cinemanage_db

Schéma simplifié :

Table : administrateurs

Champ	Type	Description
id	INT (PK, AI)	Identifiant unique
nom_utilisateur	VARCHAR(50)	Nom d'utilisateur
mot_de_passe	VARCHAR(255)	Mot de passe en clair (non chiffré)

Table : films

Champ	Type	Description
id	INT (PK, AI)	Identifiant unique
titre	VARCHAR(100)	Titre du film
realisateur	VARCHAR(100)	Réalisateur
genre	VARCHAR(50)	Genre du film
annee_sortie	INT	Année de sortie
description	TEXT	Synopsis du film

8. Déploiement et maintenance

- Le système est déployé manuellement sur le serveur via **FTP**.
- Aucun système de **gestion de versions (Git)** n'est en place.
- Les fichiers sont modifiés directement sur le serveur, sans environnement de test.
- En cas d'erreur, le service peut être interrompu.
- Les sauvegardes de la base de données sont effectuées manuellement via phpMyAdmin.
- Absence d'outils de versionnement et de collaboration.
- Faible évolutivité du système (ajout de fonctionnalités complexe).

Le système **CinéManage** actuel assure les fonctionnalités minimales nécessaires à la gestion des films et à leur affichage public.

III. Grid d'évaluation:

Évaluation 1 – Mardi 4 novembre – 100 points

Éléments de la compétence	Critères de performance	Points
FTNW-1. Analyser le projet de développement du site Web	Analyse détaillée des besoins et identification des modules à refondre ou développer	30
FTNW-2. Concevoir le visuel du site Web	2.1 Analyse adéquate des demandes. 2.2 Conception de maquettes correctes. 2.3 Respect des normes d'ergonomie. 2.4 Respect des règles de composition visuelle. 2.5 Respect des exigences du demandeur	40
FTNW-3. Préparer l'environnement de développement informatique	3.1 Installation correcte de la plateforme. 3.2 Installation correcte des logiciels et bibliothèques. 3.3 Configuration appropriée du système de gestion de versions	20
FTNW-5. Rédiger la documentation initiale	5.1 Détermination correcte des informations. 5.2 Notation claire du travail effectué	10

Évaluation 2 – Mardi 18 novembre – 100 points

Éléments de la compétence	Critères de performance	Points
FTNW-4. Programmer l'interface Web (maquettes + structure MVC)	4.1 Utilisation appropriée du langage. 4.2 Création appropriée des feuilles de styles. 4.3 Application correcte des styles. 4.4 Intégration médias et texte. 4.5 Outils de révision. 4.6 Adaptation responsive. 4.7 Utilisation système de version	40
FTNW-5. Programmer la logique applicative côté client (module Films + Auth)	5.1 Manipulation correcte des objets DOM. 5.2 Programmation des interactions. 5.3 Intégration d'animations ou widgets	30
FTNW-6. Déboguer la syntaxe du code	6.1 Utilisation efficace des outils. 6.2 Identification complète des erreurs. 6.3 Correction adéquate	20
FTNW-5. Documentation des modules développés	5.1 Détermination des informations. 5.2 Clarté de la notation	10

Évaluation 3 – Mardi 2 décembre – 100 points

Éléments de la compétence	Critères de performance	Points
00Q7-1. Créer la base de données	1.1 Analyse du modèle. 1.2 Analyse des spécifications SGBD. 1.3 Instructions SQL correctes	25
00Q7-2. Formuler des requêtes SQL (CRUD)	Lecture, insertion, modification, suppression correctes et efficaces	20
00Q7-3. Assurer la confidentialité et la cohérence des données	3.1 Techniques adéquates. 3.2 Gestion des autorisations. 3.3 Cryptage des données. 3.4 Contraintes et transactions	15
FTNY-3. Programmer la logique applicative côté serveur	3.1 Manipulation correcte des données. 3.2 Programmation sécurisée. 3.3 Authentification et autorisation. 3.4 Transformation des données en informations	25
FTNW-8. Appliquer le plan de tests fonctionnels	8.1 Identification des tests. 8.2 Réalisation complète. 8.3 Identification des erreurs et solutions. 8.4 Validation et notation claire	15

Évaluation 4 (Finale) – Mardi 16 décembre – 100 points

Éléments de la compétence	Critères de performance	Points
FTNY-4. Contrôler la qualité de l'application	4.1 Application rigoureuse des tests. 4.2 Revues de code et sécurité. 4.3 Correctifs pertinents. 4.4 Respect du suivi des problèmes et versioning. 4.5 Respect des documents de conception	30
FTNY-3. Programmer la logique applicative côté serveur (modules finaux)	3.1 Transformation des données en information. 3.2 Sécurisation. 3.3 Authentification et autorisation. 3.4 Interactions interface-utilisateur	30
FTNW-4. Programmer l'interface Web (intégration finale)	4.1 Langage approprié. 4.2 Feuilles de styles. 4.3 Intégration médias et texte. 4.4 Adaptation responsive. 4.5 Gestion version	25
FTNW-5. Rédiger la documentation technique et utilisateur	5.1 Détermination correcte de l'information. 5.2 Notation claire et complète	15