مكتب التكوين المهني وإنعاش الشغل
Office de la Formation Professionnelle et de la Promotion du Travail
Direction de la Recherche et de l'Ingénierie de la Formation
Division Conception des Examens

Examen National de Fin de Formation
Session de Juin 2026

Examen de Fin de Formation (Epreuve de synthèse)

Secteur :	**Digital et Intelligence Artificielle**	**Niveau :**	**Technicien Spécialisé**
Filière :	**Développement Digital option Web Full Stack**		

Variante	**V1**	**Durée :**	**4H00**	**Barème**	**100**

Consignes et Conseils aux Candidats :

- Lire attentivement toutes les questions avant de répondre.
- Justifier vos réponses lorsque cela est demandé et respecter les consignes de chaque exercice (Choix multiples, réponses rédigées, calculs, schémas, etc…). Toutes les réponses devront être justifiées avec le détail des calculs qui doit être indiqué sur la copie.
- Numéroter vos réponses conformément aux questions.
- Apporter un soin particulier à la présentation de votre copie.
- Relire attentivement votre copie avant de la rendre.

Document(s) et Matériel(s) Autorisés :

- Les documents ne sont pas autorisés.
- L'usage de crayons, téléphones, montres connectées ou tout autre appareil électronique est strictement interdit.
- Calculatrice simple (non programmable) autorisée.

Détail du Barème :

N° Des Dossiers	Travaux à réaliser	Barème
	Partie Théorique	
Dossier 1	Création d'une application Cloud native	6 pts
Dossier 2	Préparation d'un projet web	8 pts
Dossier 3	Approche agile	10 pts
Dossier 4	Gestion des données NoSQL	8 pts
Dossier 5	Programmation JavaScript	8 pts
	Total partie théorique	/40 points
	Partie Pratique	
Dossier 1	Gestion des données MySQL	10 pts
Dossier 2	Développement Front End	25 pts
Dossier 3	Développement Back End	25 pts
	Total partie pratique	/60 points
	Total Général	/100points

Préliminaire :

L'entreprise **ComitéManager Pro** souhaite développer une application web sécurisée dédiée à la gestion des comités opérationnels.

Un comité opérationnel est un groupe de travail constitué pour une année donnée, regroupant des collaborateurs membres, chacun disposant d'un rôle au sein du comité, et présidé par un responsable. Chaque comité fonctionne à travers des réunions planifiées au cours desquelles des décisions sont prises et consignées sous forme de comptes rendus. La participation des collaborateurs à ces réunions peut donner lieu à des absences, qui sont déclarées et associées à une réunion donnée.

L'application met en jeu les acteurs suivants :

- **Administrateur** : gère les comptes utilisateurs, les années, les collaborateurs, les comités ainsi que les membres de comités.
- **Président de comité** : collaborateur disposant d'un compte utilisateur, chargé de planifier les réunions de comité, de saisir les comptes rendus des réunions et de déclarer les absences des collaborateurs.
- **Collaborateur** : consulte les comités auxquels il est rattaché ainsi que le planning des réunions correspondantes.

Toutes les fonctionnalités de l'application nécessitent une authentification préalable.

Le système repose sur le schéma relationnel MySQL suivant :

users (**id**, name, email, password, role)
annees (**id**, libelle)
collaborateurs (**id**, nom, prenom, email, fonction, telephone, #user_id)
comites (**id**, intitule, objectif, type_comite, frequence, statut, #annee_id, #president_collaborateur_id)
membres_comite (**id**, #comite_id, #collaborateur_id, role_membre)
reunions_comite (**id**, date_reunion, heure_debut, heure_fin, mode_reunion, lien, reunion_valide, compte_rendu, #comite_id)
absences (**id**, #collaborateur_id, #reunion_comite_id, motif, justifiee)

Règles de gestion :

- Un comité est rattaché à une année, présidé par un responsable et composé de collaborateurs membres.
- Un collaborateur peut appartenir à plusieurs comités, avec un rôle défini au sein de chaque comité.
- Un comité peut organiser plusieurs réunions.
- Une réunion est présentielle ou à distance (lien obligatoire en distanciel).
- Une réunion peut être déclarée valide.
- Une absence est associée à une réunion et à un collaborateur.
- Un collaborateur ne peut être absent qu'une seule fois par réunion.

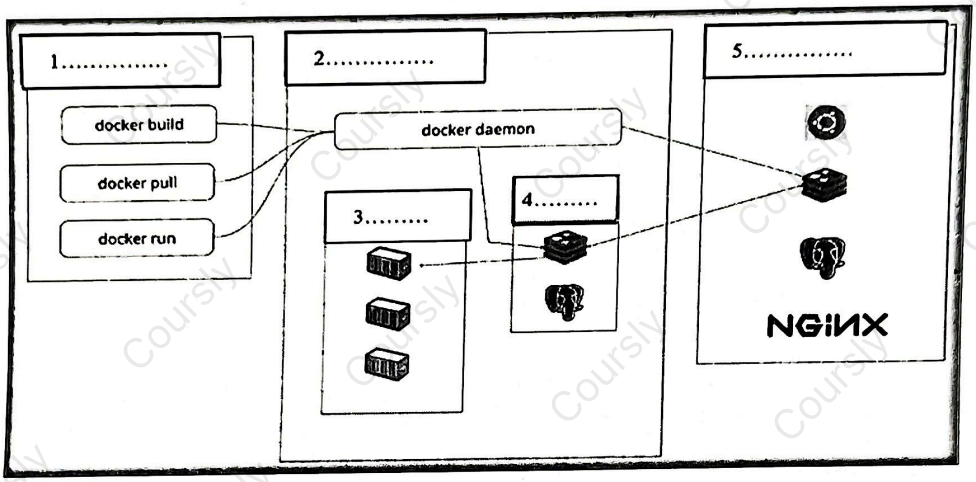
Le tableau suivant présente les valeurs possibles de certains champs du schéma relationnel.

Table	Champ	Valeurs possibles
users	role	admin, president_comite, collaborateur
collaborateurs	fonction	Technicien, Agent, Cadre, Responsable
comites	statut	Planifié, en_cours, clôturé
reunions_comite	mode_reunion	presentiel, distanciel
reunions_comite	reunion_valide	Oui, Non
absences	motif	Maladie, Congé, Autorisation, Non justifiée
absences	justifiee	Oui, Non

Partie Théorique : (40pts)

Dossier 1 : Création d'une application Cloud native (6 pts)

1. Définissez le concept d'application Cloud Native et citez trois caractéristiques techniques qui la distinguent d'une application web classique. **(2 pts)**
2. Quel est le rôle du Docker et de Kubernetes ? **(2 pts)**
3. Compléter le schéma ci dessous par les composants de l'architecture Docker. Et expliquer la différence entre une image Docker et un conteneur Docker **(2 pts)**



Dossier 2 : Préparation d'un projet web (8 pts)

Dans le cadre du projet **ComitéManager Pro,** l'entreprise souhaite modéliser les interactions entre les utilisateurs et le système avant la phase de développement de l'application web sécurisée dédiée à la **gestion des comités opérationnels**.

1. Dresser le diagramme de cas d'utilisation de l'application **ComitéManager Pro. (3 pts)**
2. Dresser le diagramme de classes de l'application **ComitéManager Pro. (3 pts)**
3. Dresser le diagramme de séquence illustrant le processus de planification d'une réunion de comité par un président de comité. **(2 pts)**

Dossier 3 : Approche Agile (10 pts)

L'équipe de développement du projet **ComitéManager Pro** adopte la méthode Scrum pour la réalisation de l'application.

1. Citer deux avantages de la méthode agile par rapport à une méthode classique de gestion de projet. **(2 pts)**
2. Définir ce qu'est une User Story, puis proposer une User Story relative au président du comité. **(2 pts)**
3. Dans le cadre du projet **ComitéManager Pro**, chaque tâche représente une étape précise du processus de développement de l'application de gestion des comités opérationnels. À partir des données de planification ci-dessus, établir le diagramme de PERT et déterminer le chemin critique du projet. **(2 pts)**

Tâche	Description
A	Recueil et analyse des besoins fonctionnels de l'application **ComitéManager Pro**
B	Installation des outils et préparation de l'environnement de développement
C	Modélisation et implémentation de la base de données relationnelle
D	Développement des fonctionnalités de l'application web
E	Tests fonctionnels et validation finale du système

Données de planification :

Tâche	Tâche antérieure	Durée (jours)
A	—	3
B	—	4
C	B	5
D	A	4
E	C, D	2

4.Définir les notions suivantes : **(3 pts)**

a) Sprint Review.

b) Sprint Retrospective.

c) Daily Scrum

5.Quel est le rôle du Product Owner **? (1 pts)**

Dossier 4 : Gestion des données NoSQL (8 pts)

Soit la collection MongoDB nommée **Comites**, contenant les documents suivants :

```
{
  "_id": 1, "intitule": "Comité Qualité", "annee": 2025, "statut": "en_cours",
  "reunions":
   [
    {
     "id_reunion": 101,      "date_reunion": "2025-05-03",      "mode_reunion": "presentiel",
     "absences":
      [
        {
          "collaborateur": { "_id": 1, "nom": "EL AMRANI", "prenom": "Rym" },
          "motif": "Maladie",
          "justifiee": "Oui"
        }
      ]
    }
   ]
}
```

1. Écrire une requête NoSQL permettant d'afficher l'intitulé des comités qui comportent au moins une réunion en mode "presentiel". **(2pts)**
2. Écrire une requête NoSQL permettant de supprimer l'absence du collaborateur « EL AMRANI Rym » de la réunion ayant l'identifiant **101**, lorsque le motif est "Maladie" **(2pts)**
3. Écrire une requête NoSQL permettant d'afficher, pour chaque comité,le nombre total d'absences non justifiées. **(2 pts)**
4. Écrire une requête NoSQL permettant de modifier le mode de la réunion d'identifiant 101 en le passant de présentiel à distanciel, et d'ajouter le champ lien avec la valeur suivante : https://meet.google.com/def **(2pts)**

Dossier 5 : Programmation JavaScript (8 pts)

1. Créer une classe JavaScript nommée **Comite** contenant les attributs suivants : **(3pts)**

   id, intitulé, typeComité, fréquence, statut*(valeur par défaut : "Planifié")*

   La classe **Comite** inclut :

   - Un constructeur permettant d'initialiser les attributs ;
   - Une méthode **Details()** retournant une chaîne de caractères décrivant les informations du comité.

2.

a) Créer deux objets à partir de la classe **Comite** : **(0.5pt)**
   - Comité 1 : Comité *Directeur*, type *Direction*, fréquence *4*
   - Comité 2 : Comité *d'Éthique*, type *Consultatif*, fréquence *2*

b) Créer un tableau **listeComites** contenant ces deux objets, puis y ajouter un troisième comité : **(0.5pt)**
   - Comité 3 : Comité *Technique*, type *Projet*, fréquence 6, statut *en_cours*

3. Créer une fonction JavaScript **afficherComites(listeComites, idElementCible)** qui prend un tableau de comités et un ID d'élément cible et qui : **(2pts)**
   - Pour chaque comité, créer une dynamique avec le texte retourné par **Details()** ;
   - Appliquer la classe CSS "Planifié" (supposée déjà crée) si le statut du comité est "Planifié" ;
   - Insèrer-les créées dans l'élément HTML cible ;

4. Soit le code HTML suivant : **(2pts)**

```
<button id="btnCharger">Charger les comités</button>
<div id="liste-comites"></div>
```

Mettre en place un gestionnaire d'événement qui, lors du clic sur le bouton btnCharger, appelle la méthode de la **(question 3)** afin d'afficher les comités.

Partie Pratique : (60pts)

Dossier 1 : Gestion de données (MYSQL) (10 pts)

A partir du schéma relationnel (voir préliminaire) :

1. Créer une fonction MySQL nommée **fn_taux_reunions_valides_comite(IN idComite INT)** permettant de calculer et de retourner le taux de réunions validées d'un comité donné, exprimé en pourcentage. **(3pts)**
2. Créer une procédure stockée nommée **sp_indicateur_collaborateur_comite (IN idComite INT, IN idCollaborateur INT, OUT nbTotalAbsences INT)** permettant de calculer, pour un collaborateur donné et un comité identifié, le nombre total d'absences non justifiées de ce collaborateur. **(4pts)**
3. Créer un trigger permettant d'empêcher l'enregistrement d'une absence lorsqu'un collaborateur est déjà déclaré absent pour la même réunion de comité. Le trigger doit afficher un message d'erreur explicite. **(3pts)**

Dossier 2 : Développement Front End (25 pts)

On souhaite développer des interfaces web React permettant de gérer les **comités opérationnels** et leurs **réunions**.
Les données sont fournies par une **API REST** et respectent les formats suivants :

Type de données	API utilisée	Exemple de résultat retourné
Comités	**Méthode** : GET **URL** : **http://localhost:3000/comites**	[{ "id": 1, "intitule": "Comité Qualité", "objectif":"Amélioration des processus", "type_comite": "Opérationnel", "frequence": "Mensuelle", "statut": "en_cours" },...]
Réunions	**Méthode** : GET **URL** : **http://localhost:3000/comites/{id}/reunions**	[{ "id": 101, "date_reunion": "2025-05-03", "mode_reunion": "presentiel", "compte_rendu": "Validation du plan d'action", "reunion_valide": "Oui",

		"absences": [{ "justifiee": "Oui" , "motif": "Maladie",collaborateur : 1}, { "justifiee": "Non" , "motif": "Non justifiée",collaborateur : 2}] },...]

1. Créer le composant **App.jsx** qui :
   - Déclare un état nommé *comites* permettant de stocker la liste des comités ; **(0.5pt)**
   - Déclare un état *loading* indiquant le chargement des données ; **(0.5pt)**
   - Déclare un état *error* pour gérer les erreurs éventuelles ; **(0.5pt)**
   - Récupère les comités depuis l'API via l'URL : **http://localhost:3000/comites; (1.5pts)**
   - Affiche un message de chargement pendant la récupération des données **;(0.5pt)**
   - Met en place le routage conformément au tableau suivant : **(2.5pts)**

Chemin	Composant	Source des données
/	ListComites.jsx	Props (comites)
/comites/:id	DetailComite.jsx	Props (comites)
/comites/:id/reunions	ListReunions.jsx	API
/comites/:id/statistiques	StatistiquesComite.jsx	API

2. Créer le composant **ListComites.jsx** permettant :
   - D'afficher, sous forme de tableau HTML, la liste des comités reçue via les **props. (2pts)**
   - Le composant doit inclure un champ de sélection (select) permettant de filtrer dynamiquement les comités par statut (planifié, en_cours, clôturé),en l'absence de filtre, l'ensemble des comités doit être affiché. **(2pts)**
   - Le composant doit également afficher le nombre total de comités affichés après filtrage et appliquer une mise en forme conditionnelle du statut *(vert pour en_cours, orange pour planifié, gris pour clôturé).* **(1.5pts)**
   - Pour chaque comité, afficher les liens d'accès au **détail du comité**, à **ses réunions** et à **ses statistiques** **(1.5pts)**

3. Créer le composant **ListReunions.jsx** permettant de récupérer l'identifiant du comité depuis l'URL, puis de charger les réunions correspondantes à partir de l'API via l'API suivante : **http://localhost:3000/comites/{id}/reunions**.
   Le composant doit afficher, sous forme de liste, la date de la réunion, le mode de réunion et le compte rendu de la réunion. **(4pts)**

4. Créer le composant **DetailComite.jsx** qui, après avoir reçu la liste des comités via les props, récupère l'identifiant du comité depuis l'URL, trouve et affiche ses informations générales (intitulé, objectif, type, fréquence, statut), puis importe et utilise le composant ListReunions.jsx pour afficher les réunions associées. **(4pts)**

5. Créer le composant **StatistiquesComite.jsx** permettant, à partir des réunions du comité récupérées depuis l'API
   GET **http://localhost:3000/comites/{id}/reunions**, de calculer et afficher : **(4pts)**
   - Le nombre total de réunions organisées,
   - Le nombre d'absences justifiées et le nombre d'absences non justifiées. (Toutes réunions confondues)

Dossier 3 : Développement Back End (25 pts)

L'application **ComitéManager Pro** est développée à l'aide de Laravel, à partir du schéma relationnel défini dans le préliminaire.

1. On souhaite permettre aux utilisateurs de l'application de se connecter :

a) Donner la commande de création du contrôleur **AuthentificationController.php. (0.5 pt)**

b) Dans le contrôleur **AuthentificationController.php,** implémenter la méthode **connect(Request $request)** permettant de valider les champs saisis*(email et password)* et d'authentifier l'utilisateur à l'aide du mécanisme d'authentification de Laravel. **(2 pts)**

c) Après une connexion réussie, rediriger l'utilisateur selon son rôle : **(1.5 pts)**

- **Le président de comité** : vers la page listant ses comités (/comites)
- **L'administrateur** : vers le tableau de bord administrateur (/admin)
- **Collaborateur** : vers la page de ses réunions (/mes-reunions)

d) Implémenter la méthode **disconnect()** qui déconnecte l'utilisateur et le redirige vers la page de login avec un message de confirmation**.(1.5 pt)**

2. On souhaite définir les modèles Eloquent nécessaires au fonctionnement de l'application.

a) Donner la commande de création du modèle **Comite. (0.5 pts)**

b) Configurer le modèle Eloquent **Comite** afin de définir le champ $fillable et les relations nécessaires conformément au schéma relationnel. **(2.5 pts)**

Note : On suppose que les autres modèles existent déjà avec leurs relations appropriées.

✓ **3.** Après authentification, le président de comité accède à la liste des comités qu'il préside pour l'année en cours.

a) Donner la commande de création du contrôleur **ComiteController.php. (0.5pt)**

b) Dans le contrôleur **ComiteController.php**, implémenter la méthode **index ()** permettant de : **(3 pts)**

- Récupérer les comités présidés par l'utilisateur authentifié ;
- Filtrer ces comités selon l'année en cours ;
- Transmettre la liste des comités à la vue **comites/index.blade.php**.

c) Afficher les comités dans la vue **comites/index.blade.php** sous forme de tableau HTML **(2.5 pts)**

- Chaque ligne du tableau doit afficher les colonnes suivantes : **intitulé**, **objectif** et **type_comite** ;
- La dernière colonne doit contenir un lien vers la gestion des réunions attachées **(Question 4)**

4. Lorsqu'un comité est sélectionné, le président peut consulter les réunions qui lui sont associées.

a) Donner la commande de création du contrôleur **ReunionComiteController.php**. **(0.5 pt)**

b) Dans le contrôleur **ReunionComiteController.php**, implémenter les méthodes suivantes :

- **index($comiteId)** : Récupérer les réunions du comité associé et retourner les résultats au format JSON **(2.5 pts)**
- **store(Request $request, $comiteId)** : Ajouter une réunion associée au comité $comiteId , et retourner une réponse JSON contenant la réunion créée avec le code *HTTP 201* **(2.5 pts)**

5. Afin d'interdire l'accès aux fonctionnalités de gestion des comités et des réunions aux utilisateurs non autorisés :

a) Donner la commande de création du middleware **PresidentComiteMiddleware.php (0.5 pt)**

b) Implémenter la méthode **handle** afin de vérifier que l'utilisateur est authentifié et que son rôle est *president_comite*. Rediriger vers la page de login avec un message d'erreur si non autorisé. **(2 pt)**

c) Dans le fichier **routes/web.php**, appliquer le middleware **PresidentComiteMiddleware.php** aux routes suivantes en utilisant un groupe de routes : **(2.5 pts)**

- GET /comites → consultation de la liste des comités
- GET /comites/{id} → affichage du détail d'un comité
- GET /comites/{comiteId}/reunions → consultation des réunions d'un comité
- POST /comites/{comiteId}/reunions → création d'une réunion pour un comité