Examen de Fin de Formation (*Session Blanche*)

Préparer par : M. DAIF Othmane

Secteur : Digital et Intelligence Artificielle

Niveau : Technicien Spécialisé

Filière : Développement Digital — Option Web Full Stack

Durée : 4 h 00 **Barème :** /100

Consignes

- Lisez la totalité du sujet et le préliminaire avant de commencer.
- Répondez à toutes les questions ; une réponse cohérente est toujours valorisée.
- Pour les exercices de calcul (PERT), la méthode correcte est prise en compte même si le résultat final est faux ; justifiez toujours votre raisonnement.
- Soignez la lisibilité de vos schémas et de votre code.

Préliminaire — Contexte du sujet

La société **« DriveNow »** gère la location de voitures à travers plusieurs agences réparties dans le pays. Chaque voiture est rattachée à une agence et peut être louée par un client pour une période donnée, moyennant un prix par jour. La société souhaite informatiser la gestion de son parc automobile, de ses clients et de ses contrats de location.

Schéma du modèle logique de données (MLD) :

Agence(id, nom, ville, telephone)

Voiture(id, matricule, marque, modele, prix_jour, kilometrage, etat, #id_agence)

Client(id, nom, prenom, telephone, email)

Location(id, #id_voiture, #id_client, date_debut, date_fin, montant_total)

NB : la clé primaire est le premier champ souligné de chaque table ; la clé étrangère est précédée du signe #.

Partie Théorique (40 points)

Dossier 1 : Création d'une application Cloud native (8 points)

1- Définir le cloud hybride et citer deux fournisseurs de services cloud. (2 pts)
2- C'est quoi un conteneur ? Quelle est la différence entre un conteneur et une machine virtuelle ? (4 pts)
3- Donner la commande Docker permettant d'arrêter un conteneur en cours d'exécution. (2 pts)

Dossier 2 : Préparation d'un projet web (6 points)

Donner le diagramme de cas d'utilisation correspondant au système suivant : (6 pts)

Le système DriveNow sera utilisé par trois acteurs : l'Administrateur, l'Agent et le Client.

- Le client peut consulter la liste des voitures disponibles et effectuer une réservation en ligne.
- L'agent peut enregistrer une location, consulter la liste des voitures et marquer une voiture comme rendue.
- L'administrateur peut faire tout ce que fait l'agent, et peut en plus gérer les voitures (ajouter, supprimer) et gérer les agences.
- L'accès au système est sécurisé : chaque utilisateur doit s'authentifier pour accéder à ses fonctionnalités.

Dossier 3 : Approche Agile et gestion de projet (7 points)

1- Citer les grandes étapes de la méthode Scrum. (2 pts)
2- Citer trois outils utilisés dans la chaîne DevOps. (1,5 pt)
3- Soit les tâches du projet DriveNow ci-dessous :

Tâche A : Étude du besoin — durée 2 jours — antériorité : aucune.

Tâche B : Conception — durée 3 jours — antériorité : A.

Tâche C : Création de la base de données — durée 2 jours — antériorité : B.

Tâche D : Développement de l'interface — durée 4 jours — antériorité : B.

Tâche E : Intégration et tests — durée 2 jours — antériorités : C et D.

Tâche F : Mise en ligne — durée 1 jour — antériorité : E.

a- Dresser le diagramme PERT en précisant les dates au plus tôt et au plus tard. (2 pts)
b- Déterminer le chemin critique et la durée minimale du projet. (1,5 pt)

Dossier 4 : Gestion de données NoSQL — MongoDB (11 points)

1- Créer la base de données « LocationDB » et la collection « voitures », puis insérer le document suivant : (3 pts)

```
{
  "_id": "v1",
  "matricule": "12345-A-6",
  "marque": "Renault",
  "modele": "Clio",
  "prix_jour": 300,
  "kilometrage": 45000,
  "etat": "Disponible",
  "agence": { "id_agence": "1", "ville": "Casablanca" },
  "locations": [
    { "_id": "l1", "date_debut": "01/06/2026", "date_fin": "05/06/2026" }
  ]
}
```

On suppose que la collection « voitures » contient un ensemble de documents. Écrire les requêtes MongoDB permettant de :

2- Afficher les voitures triées par prix par jour croissant. (2 pts)

3- Afficher le nombre de voitures de marque « Renault ». (2 pts)

4- Mettre à jour l'état de la voiture d'identifiant « v4 » à la valeur « Disponible ». (2 pts)

5- Supprimer toutes les voitures dont le kilométrage est supérieur à 200000. (2 pts)

Dossier 5 : Web dynamique PHP (8 points)

On dispose d'un tableau PHP **$voitures** où chaque élément est un tableau associatif contenant les clés **matricule**, **marque**, **prix_jour** et **etat**.

1- Écrire un script PHP qui parcourt le tableau $voitures et affiche le nombre de voitures dont l'état est « Disponible ». (4 pts)

2- Écrire un script PHP qui calcule et affiche la somme totale des prix par jour de toutes les voitures. (4 pts)

Partie Pratique (60 points)

Dossier 1 : Gestion des données MySQL (12 points)

En vous basant sur le MLD du préliminaire, écrire les scripts MySQL répondant aux questions suivantes :

1- Écrire le script de création de la table « Location ». (2 pts)

2- Modifier la table « Voiture » en ajoutant une colonne « couleur » de type VARCHAR(30). (2 pts)

3- Écrire le trigger « TR_etat » qui empêche l'insertion d'une location si la voiture concernée a l'état « En panne ». (2 pts)

4- Écrire la fonction « fct_clientLocation » qui reçoit le matricule d'une voiture et une date, et retourne l'identifiant du client qui loue cette voiture à cette date. (2 pts)

5- Écrire la procédure « P_voituresParAgence » qui affiche le nombre de voitures pour chaque agence. (2 pts)

6- Gestion des utilisateurs et des rôles : (2 pts)

a- Créer le rôle « RoleAgent ».

b- Donner les droits d'ajout, de modification et de suppression sur les tables « Voiture » et « Location » au rôle « RoleAgent ».

c- Créer l'utilisateur « karim » avec le mot de passe « 1234 ».

d- Attribuer le rôle « RoleAgent » à l'utilisateur « karim ».

Dossier 2 : Développement Front-End (24 points)

Soit l'état initial du store Redux suivant (le store, les actions et les reducers sont déjà créés ; il ne vous est pas demandé de les écrire) :

```
InitialState = {
  voiture: {
    id: 9,
    matricule: "12345-A-6",
    marque: "Renault",
    modele: "Clio",
    prix_jour: 300,
    kilometrage: 45000,
    etat: "Disponible",
    Agence: {
      id: 1,
      nom: "DriveNow Centre",
      ville: "Casablanca"
    }
  },
  Clients:    [ { id: ..., nom: ... }, { id: ..., nom: ... }, ... ],
  Voitures:   [ { id: ..., matricule: ... }, ... ],
  EtatVoiture: [ "Disponible", "Louée", "En panne" ]
};
```

1- Écrire la composante « DetailsVoiture.js » qui lit les informations de la voiture depuis le store Redux (en utilisant useSelector) et affiche : la marque, le modèle, le matricule, le prix par jour, l'état, ainsi que le nom et la ville de l'agence. (4 pts)

2- Écrire la composante « Menu.js » qui définit, à l'aide de NavLink, les liens menant aux composantes : « Détails Voiture » (/DetailsVoiture), « Liste Voitures » (/ListeVoitures) et « Nouvelle Location » (/AjouterLocation). (4 pts)

3- Écrire le code de « App.js » qui : définit le routage (BrowserRouter, Routes, Route), appelle la composante Menu.js, et intègre le Provider du store Redux. (4 pts)

4- Écrire une composante « Formulaire.js » permettant de saisir le matricule de la voiture, la date de début et la date de fin d'une location. Le clic sur le bouton « Confirmer » calcule le nombre de jours et le montant total (nombre de jours × prix par jour) puis affiche le récapitulatif sur la même page. (4 pts)

5- Initialiser la variable du state « voitures » de la composante principale App.js en appelant l'API du back-end suivante : (4 pts)

> **Méthode HTTP :** GET
>
> **URL de l'API :** http://localhost:8000/voitures
>
> **Résultat :** un tableau d'objets voiture au format JSON (id, matricule, marque, modele, prix_jour, etat).

6- Créer la composante « ListeVoitures.js » qui reçoit le tableau des voitures et l'affiche sous forme de liste à puces (matricule, marque et prix par jour de chaque voiture). (4 pts)

Dossier 3 : Développement Back-End (24 points)

1- Donner les commandes de création des fichiers de migration : « create_agences_table », « create_voitures_table », « create_locations_table ». (1,5 pt)

2- Donner le code des méthodes up() et down() du fichier de migration de la table « voitures », en définissant le champ « matricule » comme unique et le champ « etat » avec la valeur par défaut « Disponible ». (3 pts)

3- Donner la commande permettant d'appliquer les migrations. (1 pt)

4- Donner les commandes de création des modèles : Agence, Voiture, Location. (1,5 pt)

5- Donner le code du modèle « Voiture » sachant qu'une voiture appartient à une agence et qu'une voiture possède plusieurs locations (préciser la propriété fillable et les relations). (4 pts)

6- Créer le contrôleur de ressource « VoitureController » en écrivant les méthodes index (afficher toutes les voitures), store (ajouter une voiture) et destroy (supprimer une voiture). (6 pts)

7- Créer un fichier seeder pour la table « voitures » et écrire la méthode run() permettant d'ajouter une voiture avec des données de votre choix. (3 pts)

8- Donner la route de type resource permettant d'associer toutes les actions du VoitureController. (2 pts)

9- Donner le code de la méthode du modèle « Agence » permettant de récupérer la liste de toutes les voitures d'une agence donnée. (2 pts)