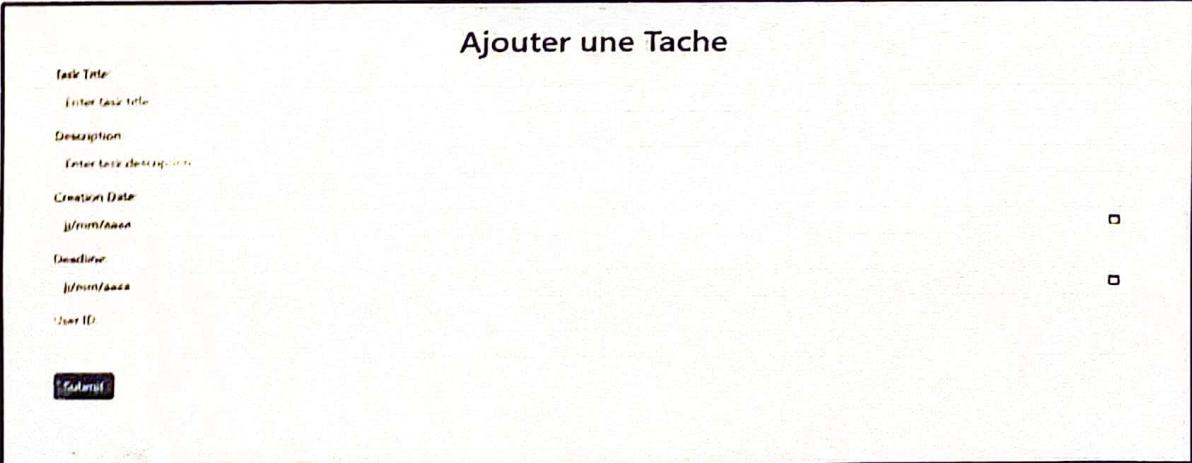


Examen de Fin de Module Régional

Module : M205 Développement back-end	Variante N° : V1	
Filière : DEVOWFS	Année : 2023/2024	Durée : 2h30 heures
Niveau : TS	Mode de formation : Résidentiel	Barème : .../40

Partie théorique (11 pts)	Note
<p>Exercice 1 : (5pts)</p> <p>Trouver la bonne réponse :</p> <p>1- Quel est le principe fondamental du concept "Migration" dans Laravel ?</p> <p>a. Gestion des sessions. b. Versionnement de la structure de la base de données. c. Configuration des routes.</p> <p>2- Quelle est la méthode utilisée pour définir une relation "Many-to-Many" dans Eloquent ?</p> <p>a. hasMany() b. belongsTo() c. belongsToMany()</p> <p>3- Quelle est la convention de nommage par défaut pour le contrôleur qui gère les opérations CRUD d'une entité "Departements" ?</p> <p>a. DepartementController. b. CRUDController c. DepartementCRUD</p> <p>4- Dans quel fichier est défini le modèle par défaut pour une table dans Laravel ?</p> <p>a. database/migrations b. app/Models c. config/database.php</p> <p>5- Quelle est la commande pour créer une nouvelle migration dans Laravel ?</p> <p>a. php artisan make:migration b. php artisan migration:make c. php artisan create:migration</p>	<p>(1 pt)</p> <p>(1 pt)</p> <p>(1pt)</p> <p>(1pts)</p> <p>(1pts)</p>
<p>Exercice 2 : (6 pts)</p> <p>Q1) Quelle est la différence entre les migrations et les seeders dans Laravel?</p> <p>Q2) Comment définir une relation "One-to-Many" entre deux modèles dans Laravel?</p> <p>Q3) Qu'est-ce que les middlewares dans Laravel et à quoi servent-ils ?</p>	<p>(2pts)</p> <p>(2pts)</p> <p>(2pts)</p>

Partie pratique : (29 pts)	Note
<p>Soit le modèle relationnel suivant :</p> <p>Utilisateurs (<u>Id</u>, nom, Email, Mot_De_Passe)</p> <p>Taches (<u>Id</u>, Titre, Description, date_De_Creation, date_De_Limit, #Utilisateur_id)</p>	
Q1) Ecrire la commande pour créer un projet laravel nommé « Gestion_des_Taches »	(1pts)
Soit une base de données vide avec le nom Gestion_des_Taches :	
Q2) Ecrire la configuration des informations de connexion de base de données (dans le fichier .env)	(2pts)
Q3) Ecrire les commandes artisan correspondant :	(1pts)
➤ Pour créer model nommé « Tache »	(1pts)
➤ Pour créer un Contrôleur de ressource nommé « TacheController »	(1pts)
➤ Pour créer une migration pour la table « Taches »	
Q4) Modifiez le fichier de migration pour définir les champs de table «Taches»	(2pts)
Q5) Ecrire la commande pour Exécutez les migrations	(1pts)
Q6) Ajouté une route de ressources pour le contrôleur « TacheController » avec un URL : «Tache»	(1pts)
Soit une vue avec le nom AjouteTache.blade.php dans le dossier /resource/views/ qui permet d'ajoute une tache à la table Taches :	
	
(Annexe 1)	
Q7) Modifier la fonction create() (dans le fichier TacheController) pour retourne la vue AjouteTache.blade.php (Annexe 1)	(1pts)
On Cliquant sur le bouton Submit form :	
Q8) Modifier la fonction store() (dans le fichier TacheController) pour Ajoute les donnée de la formulaire à la table Taches	(2pts)

Q9) Modifier `Action=""` et `Method=""` dans le formulaire de la page ci-dessus (Annexe 1) pour router la fonction `store()`

(2pts)

Soit une vue avec le nom `ListTache.blade.php` dans le dossier `/resource/views/` qui permet d'afficher la liste des Taches :

Liste des tâches						
ID	Titre	Description	Date de création	Date limite	Utilisateur ID	Actions
1	Titre de la tâche 1	Description de la tâche 1	2024-04-16	2024-04-30	1	<div>Éditer</div> <div>Supprimer</div>
2	Titre de la tâche 2	Description de la tâche 2	2024-04-16	2024-04-30	2	<div>Éditer</div> <div>Supprimer</div>

(Annexe 2)

Q10) Modifier la fonction `index()` (dans le fichier `TacheController`) pour afficher les données de la table `Taches`. Utilisez les fonctionnalités de pagination de Laravel pour afficher 10 tâches par page

(1pts)

On Cliquant sur le bouton **Modifier** (annexe 2)

Q11) Modifier la fonction `edit()` (dans le fichier `TacheController`) pour retourner une vue de modification d'une tâche existante avec un formulaire pré-rempli avec les informations de la tâche à modifier.

(4pts)

Q12) Modifier la fonction `Update()` (dans le fichier `TacheController`) pour modifier les données de la tâche à la base de données et retourner la vue `ListTache.blade.php`

(2pts)

Q13) Modifier la fonction `destroy()` (dans le fichier `TacheController`) pour supprimer une tâche et retourner la vue `ListTache.blade.php`

(2pts)

Q14) Ajoutez une vue de confirmation pour supprimer une tâche.

(1pts)

Q15) Ecrire les commandes artisan correspondant de Créer une seeder pour insérer des données de test dans la table `Taches`

(1pts)

Q16) Créez un middleware pour empêcher l'accès aux utilisateurs qui ne sont pas autorisés à modifier les tâches.

(3pts)