Nom du Module

BILAN TP DE PROGRAMMATION

Nom : Defossez
Prénom : Théo
Nom du TD/TP : Javascript-To do list

I. Instructions

Format

- Le bilan peut être **rédigé** sous forme de document écrit ou présenté sous forme de diaporama si cela est plus adapté.
- Le document final doit être soumis au format **PDF**. Assurez-vous que les fichiers sont bien nommés (évitez les espaces et utilisez des noms explicites, ex. `:bilanTP_NomPrenom.pdf`).

Clarté et Précision

- Assurez-vous que toutes les sections sont complétées avec soin. Prenez le temps de rédiger des explications claires et argumentées.
- Présentez les informations dans un ordre logique, avec des titres et sous-titres bien marqués pour faciliter la lecture.

Respect des délais

- Remettez votre bilan dans les délais impartis pour permettre une évaluation rapide et efficace. Les travaux remis en retard peuvent ne pas être pris en compte.

Étape N°1 : II. Fonctionnalités Implémentées

Liste des fonctionnalités développées et présentez des copies d'écran de votre application finale :

- Fonctionnalité 1 : Ajouter une tâche : L'utilisateur peut saisir une description de tâche dans un champ de saisie, puis cliquer sur le bouton "Ajouter une tâche" pour l'ajouter à une liste affichée dans un tableau. Chaque tâche est numérotée et ajoutée avec un bouton de suppression et une case à cocher pour indiquer son état d'achèvement.

BILAN TP DE PROGRAMMATION

- Fonctionnalité 2 : Marquer une tâche comme terminée**:** L'utilisateur peut cliquer sur une case à cocher associée à chaque tâche pour la marquer comme terminée. Lorsqu'une tâche est terminée, son libellé est barré dans le tableau.

- Fonctionnalité 3 : Supprimer une tâche**:** L'utilisateur peut supprimer une tâche spécifique en cliquant sur le bouton "Supprimer" dans la ligne correspondante du tableau. Cela met à jour automatiquement les indices des autres tâches.

Avant la suppression :

Apres la suppression :

- Fonctionnalité 4 : Filtrer les tâches**:** L'utilisateur peut filtrer les tâches affichées en fonction de leur état (toutes, terminées, non terminées) à l'aide d'un menu déroulant.

Toutes les taches :

Nom du Module

BILAN TP DE PROGRAMMATION

Seulement les taches déjà effectuer :

Seulement les taches pas encore faites :

Description détaillée :

- Chaque capture d'écran doit être accompagnée d'une explication claire et détaillée.
- Expliquez le rôle de chaque fonctionnalité (ex. : à quoi elle sert, pour qui elle est destinée).
- Précisez son fonctionnement (ex. : interaction utilisateur, logique derrière la fonctionnalité) et comment vous l'avez programmé.
- Mentionnez les problèmes rencontrés lors de son implémentation et les solutions apportées.
- Fonctionnalités visibles : pour chaque capture, précisez les fonctionnalités mises en avant (ex. : barre de navigation, formulaire, tableau de données).
- Aspects techniques : expliquez les choix techniques réalisés (ex. : technologies utilisées, structure du code, organisation des composants) et leur impact sur l'efficacité ou la performance de l'application.

Critères d'évaluation :

- Fonctionnalités listées avec précision.
- Explications approfondies et compréhensibles.
- Explications précises et bien argumentées pour chaque capture.
- Captures claires et pertinentes.
- Explications du code

Étape N°2 : III. Difficultés Rencontrées et Solutions Apportées

Difficultés :

Difficulté 1: Implémentation du filtrage des tâches **:** Lors de la mise en place du système de filtrage (par tâches terminées, non terminées, ou toutes les tâches), il y a eu un problème de performance lorsque plusieurs tâches étaient ajoutées ou modifiées. Le tableau ne se mettait pas à jour de manière fluide lors du changement de filtre, et certaines tâches restaient affichées de manière incorrecte après un filtrage.

Difficulté 2 : Dynamisme de l'interface et interactions **:** La gestion des interactions entre les éléments de l'interface (comme le bouton d'ajout, le champ de saisie, et le tableau) posait parfois des problèmes, notamment lors des premières étapes d'ajout de tâches. Par exemple, le bouton d'ajout ne s'affichait pas immédiatement après la création de l'élément, ou certaines actions comme la suppression ou le marquage d'une tâche n'étaient pas prises en compte si la page était rechargée.

Solutions :

Difficulté 1 : Implémentation du filtrage des tâches**:** Après avoir analysé le problème, j'ai ajusté la logique de la fonction `filterTasks()` pour mieux gérer l'affichage dynamique. J'ai modifié le code afin que le filtrage ne soit effectué qu'après que toutes les modifications d'état aient été appliquées. J'ai également veillé à ce que la fonction masque correctement les lignes du tableau en fonction de l'état de chaque tâche, et cela a permis d'optimiser les performances du filtrage.

Difficulté 2 : Dynamisme de l'interface et interactions **:** Pour régler ce problème, j'ai utilisé des événements (`addEventListener`) de manière plus systématique et claire, en les attachant aux bons éléments dès leur création dynamique dans le DOM. J'ai également intégré des vérifications pour m'assurer que les événements étaient correctement liés à chaque bouton ou case à cocher, ce qui a amélioré la réactivité de l'interface.

Critères d'évaluation :

- Difficultés bien identifiées et pertinentes.
- Solutions claires, adaptées et bien expliquées.

Étape N°3 : IV. Commentaires et Suggestions

Remarques générales

Ce qui a bien fonctionné **:** Le développement de l'application dans son ensemble s'est bien passé, en particulier la mise en place des fonctionnalités de base comme l'ajout, la suppression et le marquage des tâches. Les fonctionnalités essentielles ont été bien intégrées, et le code a fonctionné comme prévu après avoir résolu les premiers bugs. Le système de gestion des tâches est assez flexible et a permis une personnalisation en fonction des besoins spécifiques.

Suggestions d'amélioration :

Gestion des erreurs et feedback utilisateur **:** Lorsqu'une tâche est ajoutée ou supprimée, il serait utile d'ajouter un feedback visuel pour l'utilisateur, comme une notification qui confirme l'action (ajout ou suppression réussie). Cela améliorerait l'expérience utilisateur et réduirait la confusion possible lorsqu'il n'y a pas de message explicite pour indiquer qu'une tâche a été correctement ajoutée ou supprimée.

Focus sur les Apprentissages

Sur le plan technique : J'ai appris à mieux gérer l'interaction dynamique entre le DOM et JavaScript, notamment pour manipuler des éléments d'interface tels que les tableaux et les cases à cocher. En particulier, la gestion des événements et des changements d'état des tâches m'a permis d'approfondir mes connaissances sur les fonctions de rappel (`eventListeners`) et leur gestion. La difficulté d'affichage lors du filtrage m'a également appris à mieux optimiser l'interaction avec les éléments du DOM pour éviter des ralentissements ou des comportements inattendus.

Sur le plan méthodologique : La gestion des erreurs dans le filtrage des éléments et la réactivité du tableau a mis en lumière l'importance de tester et de déboguer chaque étape de développement. Utiliser des console logs pour analyser le comportement du filtrage et des événements m'a permis de mieux cerner le problème. Ce type de travail m'a également appris à être plus vigilant sur l'optimisation du code, ce qui sera utile dans des projets plus complexes à l'avenir.

Critères d'évaluation :

- Observations pertinentes et réflexion approfondie.
- Suggestions concrètes et réalistes.

Étape N°4 : V. Auto-évaluation

Évaluation de votre travail :

Planification : 7/10 La planification a été assez simple, car le projet était relativement basique. Cependant, je n'ai pas pris suffisamment de temps pour définir les différentes étapes de développement en détail, ce qui m'a fait perdre un peu de temps lorsque des problèmes sont apparus.

Codage : 8/10 Le codage a été globalement fluide. J'ai réussi à implémenter les fonctionnalités principales rapidement (ajout, suppression, filtrage). Cependant, j'ai rencontré quelques petites difficultés avec le filtrage et la gestion des tâches marquées comme terminées, mais elles ont été résolues après quelques ajustements.

Tests : 6/10 Les tests ont principalement été faits manuellement en testant les fonctionnalités au fur et à mesure. Je n'ai pas mis en place de tests automatisés.

Critères d'évaluation :

- Auto-évaluation objective et bien argumentée.
- Objectifs pertinents et réalistes.

Étape N°5 : VI. Code

Lien GitHub :

- Ajoutez un lien vers votre dépôt GitHub et vérifiez qu'il est accessible.
- Organisez votre dépôt en plusieurs dossiers (ex. : `src` pour le code source, `images` pour les images et ressources, `docs` pour la documentation ….). Assurez-vous que chaque dossier a une utilité claire et est bien structuré.

README :

- Incluez un fichier README complet et structuré contenant :
  - Une description du projet.
  - Les instructions d'installation et d'utilisation.
  - Les prérequis et outils utilisés.

Critères d'évaluation :

- Lien GitHub fonctionnel.
- README bien élaboré.
- Code propre, bien organisé et documenté