

Énoncé du Projet : Application TodoList avec Redux + Workflow Git/GitHub Collaboratif

Objectif général

Créer une application **TodoList** en **React + Redux (et Redux Toolkit)** permettant d'ajouter, modifier et supprimer des tâches.

Le projet doit être réalisé **en groupe**, en utilisant les **bonnes pratiques Git/GitHub** :

- **Fork**
- **Clone**
- **Branches** pour chaque fonctionnalité
- **Pull Requests**
- **Merge**
- **Pull pour synchroniser**
- **Résolution de conflits**

Fonctionnalités obligatoires

Votre application TodoList doit contenir :

1. Ajouter une tâche
 - Champ texte + bouton “Add”
 - Chaque tâche contient :
 - `id`
 - `title`
 - `completed` (false par défaut)
2. Supprimer une tâche
3. Modifier une tâche
4. Marquer une tâche comme faite / non faite (completed)
5. Filtrer les tâches
 - All
 - Completed
 - Not completed
6. Design simple mais propre en utilisant bootstrap

Structure Redux obligatoire

Vous devez créer :

- Un store Redux
- Un reducer todosReducer

Avec les actions suivantes :

- ADD_TODO
- DELETE_TODO
- UPDATE_TODO
- TOGGLE_TODO

Pas de Redux Toolkit.

État initial : [].

GitHub : Exigences de collaboration

L'équipe doit suivre **strictement** le workflow suivant.

Étape 1 — Création du dépôt original

Un membre de l'équipe :

1. crée un dépôt GitHub nommé :
todolist-redux-project
2. ajoute un README
3. crée la structure initiale React :

```
npx create-react-app todolist-redux
```

Il pousse tout vers GitHub (git push origin main).

Étape 2 — Fork du projet

Chaque membre de l'équipe doit :

1. **Forker le dépôt original**
2. Cloner SON fork localement :

```
git clone https://github.com/<username>/todolist-redux-project.git
```

3. Ajouter le dépôt original comme **remote upstream** :

```
git remote add upstream https://github.com/<owner>/todolist-redux-project.git
```

Étape 3 — Travail avec branches

Chaque fonctionnalité doit être faite dans **une nouvelle branche** :

Exemples de branches obligatoires :

Fonctionnalité	Nom de la branche
Configuration Redux	setup-redux
Ajouter une tâche	feature/add-todo
Supprimer une tâche	feature/delete-todo
Modifier une tâche	feature/update-todo
Toggle done	feature/toggle-todo
Filtre	feature/filter-todos

Créer une branche :

```
git checkout -b feature/add-todo
```

Étape 4 — Pull Request

Quand une fonctionnalité est terminée :

1. pousser la branche :

```
git push origin feature/add-todo
```

2. Aller sur GitHub → **Créer une Pull Request** vers main
3. Un autre membre doit :
 1. lire la PR
 2. commenter
 3. approuver
 4. merger vers main

Étape 5 — Récupérer les mises à jour

Chaque membre doit rester synchronisé :

```
git pull upstream main  
git push origin main
```

Livrables attendus

1. Lien du dépôt original
2. Lien du dépôt forké de chaque membre

3. Liste des PR créées
4. Capture d'écran de l'application