République du Cameroun

Paix-travail-patrie

Ministère de l'enseignement supérieur

Université de Maroua

Ecole nationale supérieure Polytechnique

Republic of Cameroon

Peace-work-fatherland

Ministry of higher education

The university of Maroua

National Advanced school of engineering

CAHIER DES CHARGES

ITE 316 (Frameworks et IDE Web)

Thème : Application de Gestion de Tâches avec React JS

N	NOMS ET PRENOMS	MATRICULES
1	BOUNYAMINE OUSMANOU	24ENSPM445
2	DIYEN YEM BRIAN	22ENSPM446
3	KAMDEM DEFO BAKAM ONESIME	22ENSPM474
4	NOUGA MFANGNIA ANGE MERVEILLE	24ENSPM340

Niveau et filière : IC-3 Informatique et télécommunication (INFOTEL)

Option : Génie Logiciel 3 (Groupe 1)

Examinateur : M MANAODA

Année académique : 2024-2025

Table des matières

1. INTRODUCTION

1.1 Objectifs du projet

Ce document constitue le Cahier de charges pour une application web moderne de gestion de tâches permettant aux utilisateurs de créer, organiser et suivre l'avancement de leurs activités. L'objectif principal est de concevoir une interface utilisateur intuitive et réactive utilisant React JS pour consommer une API de gestion de tâches.

1.2 Contexte

Dans le cadre d'un projet académique, cette application servira à démontrer la maîtrise des technologies front-end modernes, particulièrement React JS. L'application simule un environnement de travail collaboratif où plusieurs utilisateurs peuvent gérer des tâches communes.

1.3 Portée du projet

Le projet couvre le développement complet du front-end d'une application de gestion de tâches, incluant :

- L'interface utilisateur responsive
- L'intégration avec une API existante
- La gestion des états et des données
- Les fonctionnalités de base CRUD (Create, Read, Update, Delete)

2. SPÉCIFICATIONS FONCTIONNELLES

➢ Spécification Fonctionnelle

2.1 Fonctionnalités principales

2.1.1 Gestion des tâches

Création de tâches :

- Formulaire de création avec champs obligatoires (titre, description, nom_utilisateur)
- Champs optionnels (date d'échéance, priorité, statut)
- Validation des données en temps réel
- Confirmation de création avec notification

Modification de tâches :

- Édition en ligne des propriétés de la tâche

Suppression de tâches :

- Confirmation avant suppression

2.1.2 Assignation des tâches

Gestion des utilisateurs simulés :

- Liste prédéfinie d'utilisateurs fictifs
- Profils avec nom, email et avatar

Processus d'assignation :

- Sélection d'utilisateur via menu déroulant ou recherche
- Notification visuelle de l'assignation

2.1.3 Suivi de l'avancement

Statuts des tâches :

- À faire (Todo)
- En cours (In Progress)
- Terminée (Completed)

Marquage comme terminée :

- Action simple par clic/bouton

2.2 Fonctionnalités secondaires

- Filtrage des tâches par statut, assigné, date
- Recherche textuelle dans les tâches
- Tri par différents critères

➢ Spécifications non fonctionnelles

- Assurer l'évolutivité
- Assurer l'ergonomie
- Accecibilite
- Flexibilité

3. SPÉCIFICATIONS TECHNIQUES

3.1 Architecture logicielle

Architecture Front-end :

- Architecture en composants React
- Gestion centralisée de l'état avec Contexte API ou Redux
- Structure modulaire et réutilisable

Pattern architectural :

- Composants conteneurs et composants de présentation
- Services pour l'interaction avec l'API
- Hooks personnalisés pour la logique métier

3.2 Technologies utilisées

Front-end :

- React JS 18.x
- React Router pour la navigation
- Axios pour les appels API
- CSS Modules (Tailwind CSS)
- Material-UI pour l'interface

Objectif du choix de React-js

- **Création des interfaces utilisateurs (UI) Reactive et interactive :**

- Permet aux utilisateurs de créer, modifier des taches en temps réel sans rechargement de page ;
- Offrir une expérience fluide lors du filtrage, recherche ou tri des taches ;
- Assurer des notifications visuelles (exemple : confirmation de création, suppression, changement de statut)

Outils de développement :

- React + Vite
- Git pour le versioning
- Visual Studio Code comme éditeur de code

API :

- API REST JSON
- Authentification par token JWT (simulée)
- Endpoints CRUD pour les tâches et utilisateurs

4. INTERFACES UTILISATEUR

4.1 Design général

Principes de design :

- Design responsive (mobile-first)

- Interface claire et intuitive
- Cohérence visuelle
- Accessibilité (WCAG 2.1)

4.2 Écrans principaux

4.2.1 Écran principal - Liste des tâches

Éléments visuels :

- Header avec titre et boutons d'action
- Zone de filtres et recherche
- Liste des tâches avec cards
- Bouton flottant pour création rapide

Interactions :

- Clic sur tâche pour voir détails
- Actions rapides (compléter, assigner)

4.2.2 Formulaire de création/édition

Disposition :

- Formulaire modal ou page dédiée
- Champs organisés logiquement
- Validation en temps réel
- Boutons d'action clairement identifiés

4.2.3 Vue détaillée d'une tâche

Contenu :

- Informations complètes de la tâche
- Historique des modifications
- Commentaires (si applicable)
- Actions disponibles

4.3 Composants réutilisables

- Boutons avec états (loading, disabled)
- Formulaires avec validation
- Modales et notifications
- Indicateurs de statut et priorité

5. DÉPLOIEMENT

5.1 Environnements

Développement :

- Serveur local (localhost :5173)
- API de développement
- Données de test

5.2 Configuration

Variables d'environnement :

- URL de l'API
- Clés d'authentification
- Configuration des services tiers

6. PLANNING

6.1 Phases du projet

Phase 1 : Analyse et conception (1 jour)

- Finalisation des spécifications
- Création des maquettes
- Architecture technique détaillée

Phase 2 : Développement core (2 jours)

- Développement des composants de base
- Intégration API pour les tâches
- Interface de création/édition

Phase 3 : Fonctionnalités avancées (2 jours)

- Système d'assignation
- Gestion des statuts
- Filtres et recherche
- Interface responsive

6.2 Jalons principaux

- **J+1** : Spécifications validées et maquettes approuvées
- **J+2** : Première version fonctionnelle (CRUD basique)
- **J+2** : Version complète avec toutes les fonctionnalités

6.3 Livrables

- Code source commenté et documenté
- Documentation technique
- Guide utilisateur
- Application déployée et fonctionnelle

7. Budget

Poste	Coût estimé
Hébergement (GitHub Pages)	Gratuit
Logiciels (React, JSON Server, outils de dev (Visual Studio Code)	Gratuit
Temps de développement (étudiant)	Non rémunéré
Connexion internet	5000 FCFA
Total estimé	**5000 FCFA**

CONCLUSION

Ce cahier des charges définit les spécifications complètes pour le développement d'une application de gestion de tâches moderne utilisant React JS. Le projet, prévu sur 6 jours pour un budget de 5000F permettra de livrer une solution fonctionnelle répondant aux besoins exprimés.

La réussite du projet repose sur le respect du planning, la qualité du développement et une communication efficace entre tous les acteurs du projet.