

EMSI

Filiere : Informatique / Genie Logiciel

# Application de Gestion de Taches

*To-Do List Mobile React Native*

## Rapport de Projet

Annee academique 2025–2026

### Sujet :

Developpement d'une application mobile de productivite avec React Native, gestion avancee des taches et tableau de bord de statistiques

### Realise par :

ANOUAR GACHOU  
ADNANE BELIYACHI  
ADIB ZINDINE

### Encadrant :

ID : SAADI MOSTAFA

Date : 13 janvier 2026

## Résumé

Ce rapport presente une application mobile de gestion de taches developpee en React Native et Expo. L'application integre des fonctionnalites avancees : gestion des priorites, categories, sous-taches, dates d'echeance, et un tableau de bord de productivite avec statistiques. Le document couvre l'architecture technique, les fonctionnalites principales, et les diagrammes UML de conception.

**Mots-cles :** React Native, Expo, TypeScript, Gestion de Taches, UML, AsyncStorage, Architecture Logicielle

# Table des matières

<b>1</b>	<b>Introduction et Objectifs</b>	<b>3</b>
1.1	Objectifs du Projet . . . . .	3
1.2	Stack Technologique . . . . .	3
<b>2</b>	<b>Architecture et Fonctionnalités</b>	<b>3</b>
2.1	Gestion des Tâches . . . . .	3
2.2	Tableau de Bord . . . . .	3
2.3	Architecture Logicielle . . . . .	4
<b>3</b>	<b>Diagrammes de Conception</b>	<b>6</b>
3.1	Diagramme de Cas d'Utilisation . . . . .	6
3.2	Diagramme de Flux . . . . .	8
3.3	Diagramme de Classes . . . . .	9
3.4	Diagramme de Séquence . . . . .	9
<b>4</b>	<b>Implementation Technique</b>	<b>9</b>
4.1	Modèle de Données . . . . .	9
4.2	Opérations CRUD . . . . .	10
<b>5</b>	<b>Qualités et Améliorations</b>	<b>10</b>
5.1	Points Forts . . . . .	10
5.2	Opportunités d'Optimisation . . . . .	10
<b>6</b>	<b>Roadmap Futur</b>	<b>11</b>
6.1	Court Terme (0-3 mois) . . . . .	11
6.2	Moyen Terme (3-6 mois) . . . . .	11
6.3	Long Terme (6+ mois) . . . . .	11
<b>7</b>	<b>Conclusion</b>	<b>11</b>

# 1 Introduction et Objectifs

L'application To-Do React Native est une solution de productivite personnelle pour appareils mobiles (iOS et Android). Contrairement aux To-Do lists minimalistes, elle integre des fonctionnalites professionnelles : priorites, categories, sous-taches, dates d'echeance et tableau de bord analytique.

## 1.1 Objectifs du Projet

1. Fournir une UX moderne et intuitive avec animations fluides
2. Implementer des fonctionnalites avancees de gestion des taches
3. Garantir un fonctionnement complet hors ligne via AsyncStorage
4. Concevoir une architecture maintenable et evolutive
5. Proposer un tableau de bord motivant avec statistiques

## 1.2 Stack Technologique

Technologies principales :

- **React Native v0.81.5** : Framework mobile cross-platform
- **Expo v54.0.30** : Plateforme de developpement
- **TypeScript** : Langage avec typage statique
- **Expo Router** : Navigation par fichiers
- **AsyncStorage** : Persistance locale des donnees

# 2 Architecture et Fonctionnalites

## 2.1 Gestion des Taches

Le systeme permet :

- **CRUD complet** : Creation, lecture, mise a jour, suppression
- **Priorites** : Haute (rouge), Moyenne (orange), Basse (vert)
- **Categories** : Travail, Personnel, Maison, Autres
- **Sous-taches** : Checklists integrees pour decomposer les taches
- **Dates d'echeance** : Saisie via DateTimePicker natif
- **Interactions** : Swipe actions, filtres, recherche en temps reel

## 2.2 Tableau de Bord

Le tableau de bord affiche :

- Nombre de taches terminees
- Taux de reussite (pourcentage)

- Taches restantes
- Taches haute priorite
- Messages motivationnels dynamiques

## 2.3 Architecture Logicielle

L'application suit le principe de **separation des preoccupations** :

- **Logique metier** : Operations CRUD sur les taches
- **Interface utilisateur** : Composants React personnalisés
- **Persistence** : AsyncStorage pour stockage local

Le typage **TypeScript exhaustif** garantit la robustesse du code. Les **Hooks React** (useState, useEffect, useCallback, useMemo) optimisent les performances.



## 3 Diagrammes de Conception

### 3.1 Diagramme de Cas d'Utilisation

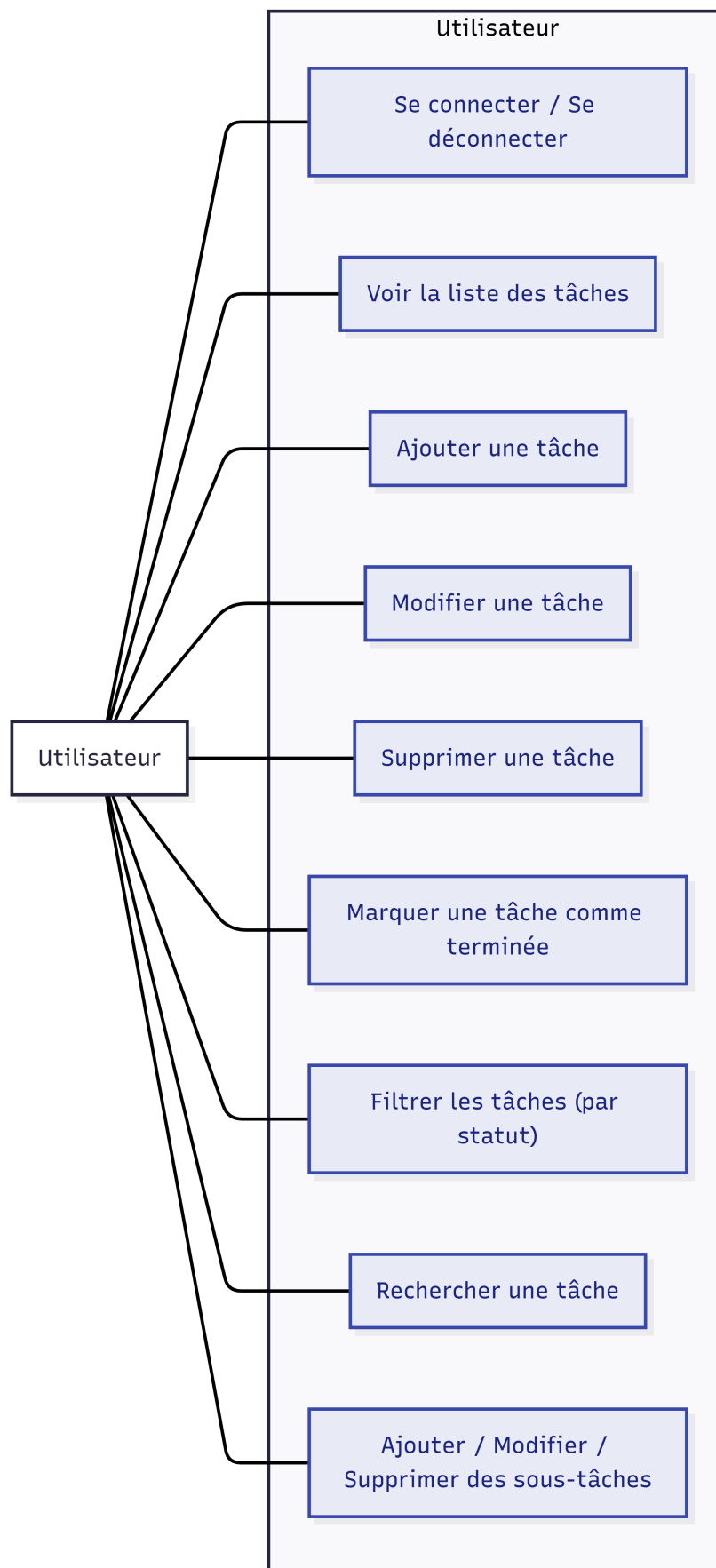
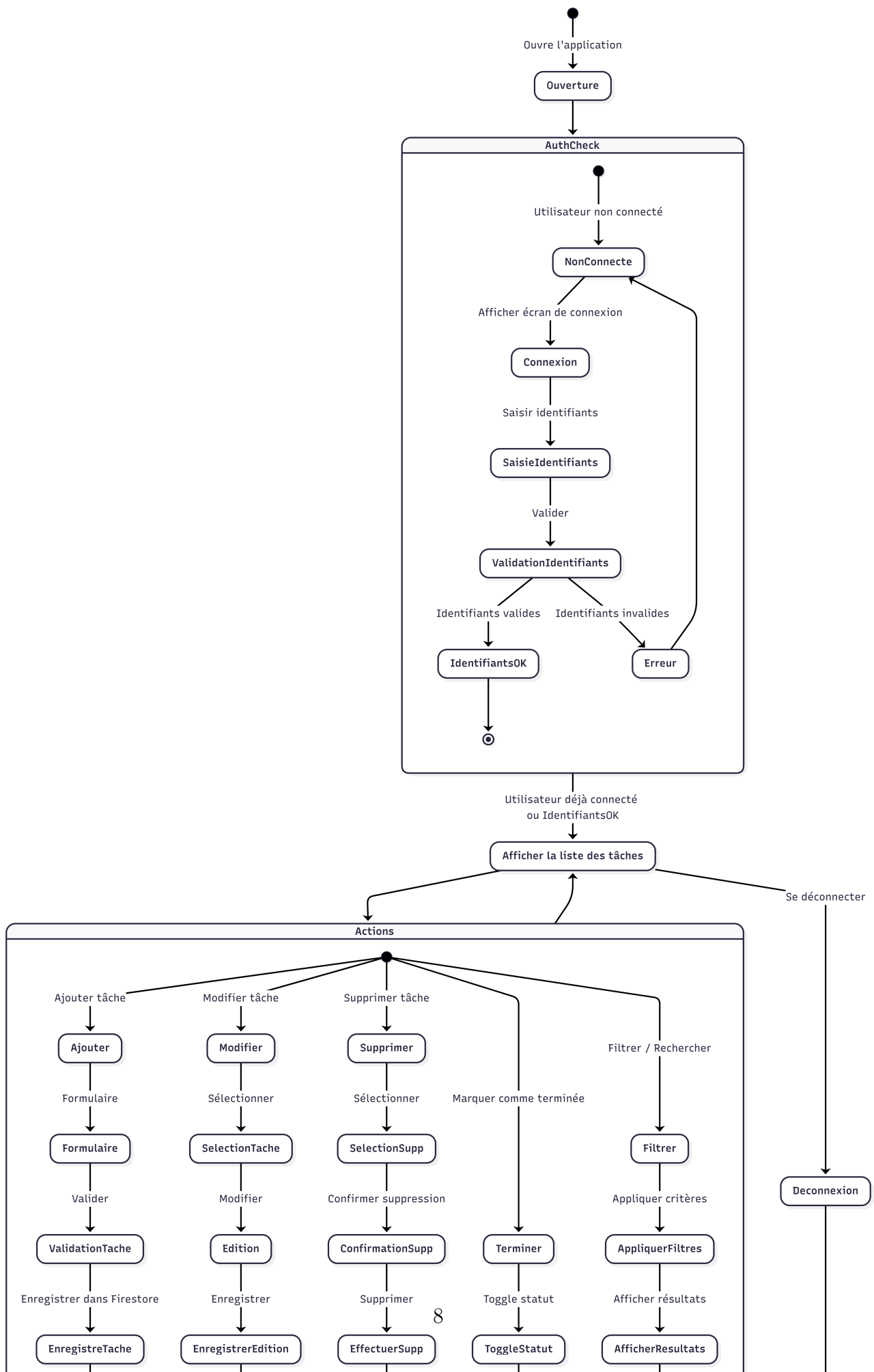


FIGURE 1 – Fonctionnalités offertes à l'utilisateur





## 3.2 Diagramme de Flux



### 3.3 Diagramme de Classes

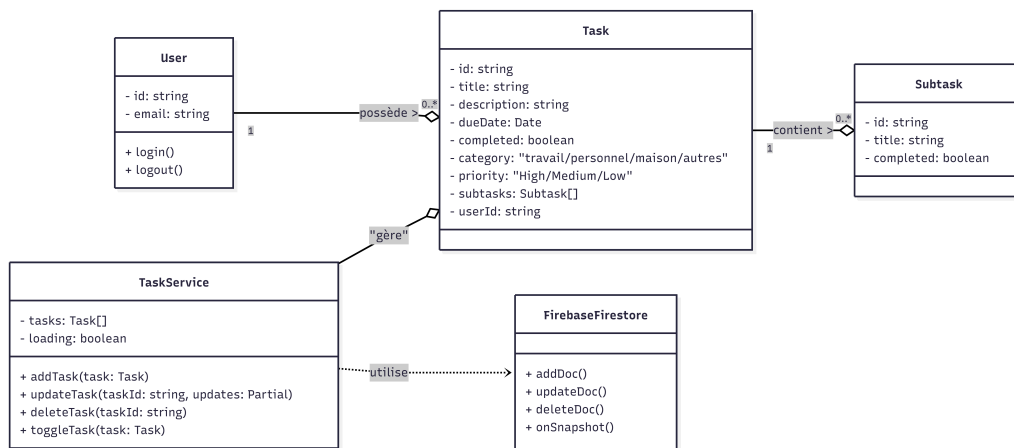


FIGURE 3 – Structure du modèle métier (Utilisateur, Tache, GestionTaches)

### 3.4 Diagramme de Sequence

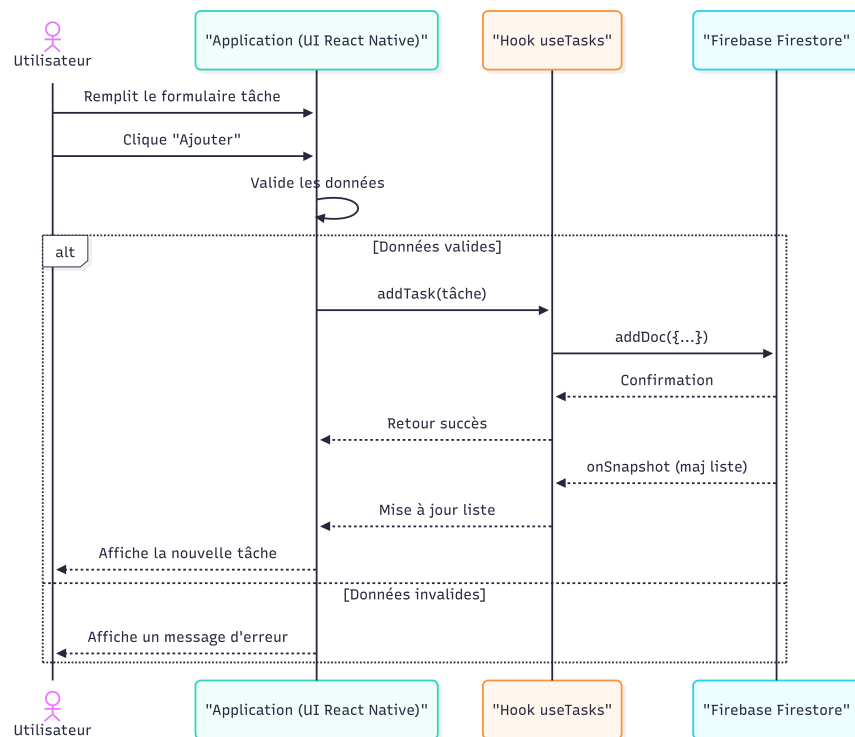


FIGURE 4 – Scenario d'ajout d'une tâche : validation et sauvegarde

## 4 Implementation Technique

### 4.1 Modèle de Données

Listing 1 – Interfaces TypeScript principales

```
1 interface Task {
2   id: string;
3   title: string;
4   priority: 'Haute' | 'Moyenne' | 'Basse';
5   category: 'Travail' | 'Personnel' | 'Maison';
6   dueDate?: Date;
7   completed: boolean;
8   subtasks: Subtask[];
9 }
10
11 interface Subtask {
12   id: string;
13   title: string;
14   completed: boolean;
15 }
```

## 4.2 Operations CRUD

Listing 2 – Exemple : Suppression de tache

```
1 const deleteTask = useCallback((taskId) => {
2   const updated = tasks.filter(t => t.id !== taskId);
3   setTasks(updated);
4   AsyncStorage.setItem('tasks',
5     JSON.stringify(updated));
6 }, [tasks]);
```

## 5 Qualites et Améliorations

### 5.1 Points Forts

#### Forces majeures :

- Architecture moderne avec Expo Router et Hooks React
- Typage TypeScript complet = robustesse et maintenabilite
- UX sophistiquee : animations, swipe, feedback haptique
- Autonomie hors ligne via AsyncStorage
- Tableau de bord analytique avec motivation utilisateur

### 5.2 Opportunités d'Optimisation

1. **Refactorisation** : Fichier principal `index.tsx` (800+ lignes) → extraire composants (TaskItem, TaskForm, TaskFilters)

2. **Gestion d'état centralisée** : Migration vers **Zustand** pour cohérence entre écrans
3. **Tests automatisés** : Suite complète pour qualité long terme
4. **Virtualisation des listes** : FlatList pour listes longues

## 6 Roadmap Futur

### 6.1 Court Terme (0-3 mois)

Tâches récurrentes, tags personnalisés, notes avec formatage riche, notifications locales.

### 6.2 Moyen Terme (3-6 mois)

Synchronisation cloud (Firebase), partage de tâches, intégration calendrier, graphiques de productivité.

### 6.3 Long Terme (6+ mois)

Suggestions IA, intégrations tierces (Google, Notion, Slack), version web, mode collaboratif.

## 7 Conclusion

L'application To-Do React Native offre une **gestion de tâches avancée** avec architecture moderne, typage fort TypeScript, et UX soignée. Les diagrammes UML confirment la cohérence de la conception. Avec les améliorations proposées (refactorisation, Zustand, tests), le projet a le potentiel de devenir une solution mobile leader de productivité.

Le rapport, la page de garde personnalisée et les diagrammes constituent un livrable complet destiné à mettre en valeur le travail de l'équipe.

— *Fin du Rapport* —