Implémentation d'une application de gestion de tache

M. FOKOU KEMEGNE - Encadreur Académique

M. HANNIEL TSASSE - Encadreur Professionel(FAROTY S.A.S)

NKWE AHOUME BORIS - Etudiant en Genie Logiciel

nkabo074@gmail.com

July 15, 2024

Dedicace

À mes parents, qui m'ont toujours soutenu et encouragé à poursuivre mes rêves. Merci de croire en moi.

Remerciment

Je souhaite exprimer ma profonde gratitude à toutes les personnes qui ont contribué à la réalisation de ce mémoire. Tout d'abord, je remercie Dieu Tout-Puissant pour m'avoir accordé la force, la sagesse et la persévérance nécessaires pour mener à bien ce projet. Je tiens à remercier chaleureusement:

- le Directeur de l'Institue Universitaire de technologie de Douala pour le soutien institutionnel et les opportunités académiques offertes tout au long de mon parcours. Mes remerciements vont également au
- Responsable Académique M. MBAI, dont les conseils avisés et le dévouement ont été précieux.
- Mes parents, pour leur amour inconditionnel, leur soutien moral et financier, et leurs encouragements constants qui ont été une source inestimable de motivation.
- Mes frères et surs, je suis reconnaissant pour leur soutien affectueux et leur compréhension durant les moments difficiles.
- Mon Encadreur Professionnel M. HANNIEL TSASSE, dont l'expertise et les conseils pratiques m'ont guidé tout au long de cette recherche.
- Mon Encadreur Académique M. FOKOU KEMEGNE mérite aussi une reconnaissance particulière pour son accompagnement pédagogique, sa patience et ses précieux conseils.

Un grand merci à Je remercie également

mes professeurs, qui ont partagé leur savoir et leur passion, contribuant ainsi de manière significative à ma formation.

merci sincère à mes camarades, pour leur soutien, leur camaraderie et les moments partagés qui ont rendu cette expérience mémorable et enrichissante.

À vous tous, merci infiniment.

Somaire

Résumé

Le présent rapport détaille la conception et le développement d'une application de gestion de tâches, nommée GETDO pour le compte de la société FAROTY SAS. L'objectif principal de cette application est d'aider les utilisateurs à organiser efficacement leurs tâches quotidiennes, à suivre leur progression et à améliorer leur productivité. L'application propose plusieurs fonctionnalités clés, notamment l'assignation de tâches avec des dates d'échéance, un tri en fonction des priorités, des rappels,un calendrier, un suivi de l'accomplissement des tâches, un tableau de vision pour visualiser les objectifs à long terme, ainsi qu'un tableau de bord offrant une vue d'ensemble de la progression des tâches à l'aide de graphiques. Le développement de l'application s'est appuyé sur des technologies modernes telles que Vue JS pour le front-end et FireBase pour le back-end, offrant ainsi une expérience utilisateur fluide et intuitive. De plus,l'application fonctionnalités avancées telles que une optimisation de la planification et une analyse de productivité. Ce rapport décrit en détail le processus de conception, de développement et de test de l'application, ainsi que les défis rencontrés et les solutions apportées. En outre, il met en évidence les perspectives d'amélioration et les pistes pour de futures recherches et développements dans le domaine de la gestion de tâches et des possibilité de l'intégration de l'Intelligence Artificielle. En conclusion, l'application GETDO représente une contribution significative dans le domaine de la gestion de tâches, offrant aux utilisateurs une solution moderne et efficace pour gérer leur travail et améliorer leur productivité au quotidien.

Liste des figures

Introduction Générale

Dans un monde où le temps et les ressources sont des éléments précieux, la gestion efficace des tâches et des projets revêt une importance capitale pour les individus et les organisations. Dans ce contexte, les applications de gestion de tâches jouent un rôle essentiel en offrant des outils et des fonctionnalités permettant de planifier, suivre et organiser les activités quotidiennes et les projets à long terme de manière efficace et méthodique. Le présent rapport détaille le processus de conception et de développement d'une application de gestion de tâches moderne, intitulée GETDO. Notre objectif principal est de fournir aux utilisateurs une solution robuste et intuitive pour gérer leurs tâches personnelles et professionnelles, en mettant l'accent sur l'optimisation de la productivité, la facilitation de la collaboration et l'amélioration de l'efficacité opérationnelle. Ce rapport présentera en détail les différentes phases du projet, depuis la phase de conception initiale jusqu'au déploiement final de l'application. Nous aborderons les aspects techniques, fonctionnels et conceptuels du développement, en mettant en lumière les défis rencontrés, les solutions apportées et les choix technologiques effectués tout au long du processus. De plus, nous explorerons les opportunités offertes par l'intégration de l'intelligence artificielle dans notre application, en examinant comment les fonctionnalités avancées telles que les recommandations

intelligentes, l'analyse de données et l'automatisation peuvent contribuer à améliorer l'expérience utilisateur et à optimiser la gestion des tâches. Enfin, nous discuterons des perspectives d'avenir pour notre application, en identifiant les pistes pour de futures améliorations, les évolutions technologiques envisagées et les possibilités d'extension et d'adaptation pour répondre aux besoins changeants des utilisateurs et du marché. À travers ce rapport, nous espérons fournir un aperçu complet de notre travail et de notre vision pour l'application GETDO, ainsi que de son potentiel à contribuer à l'efficacité et à la réussite des individus et des organisations dans leur gestion quotidienne des tâches et des projets.

Chapitre 1

Présentation de l'entreprise FAROTY SAS

FAROTY SAS est une société camerounaise innovante qui offre des solutions financières numériques à une clientèle grandissante. Localiser a près de lhôtel Kremlin a Bonamousadi, FAROTY a pour mission de faciliter l'accès aux services financiers de base pour tous les Camerounais, en particulier ceux des zones rurales et non bancarisées. FAROTY propose une suite complète de solutions financières numériques, notamment :

Paiements et transferts d'argent : Envoyez et recevez de l'argent facilement et en toute sécurité via des plateformes de paiement mobile et électronique (Orange Money, MTN Mobile Money, PayPal).

Gestion financière personnelle : Gérez vos finances personnelles en toute simplicité avec nos outils de budgétisation, de suivi des dépenses et d'épargne.

Collecte de fonds (Crowdfunding) : Levez des fonds pour vos projets

et causes grâce à notre plateforme de crowdfunding sécurisée.

Tontines digitales : Participez à des tontines virtuelles et gérez vos cotisations en ligne.

Billetterie électronique : Achetez et vendez des billets d'événements en ligne avec notre service ePass.

Pages commerciales : Créez votre propre boutique en ligne et vendez vos produits et services facilement. Leurs technologie

FAROTY a un impact positif sur les Camerounais en leur offrant un accès plus facile aux services financiers il est donc de notre devoir de facilité leurs travails via l'application présenter dans ce rapport

Chapitre 2

Méthode et equipements utilisé

2.1 équipement et logiciel utiliser

Pour mener à bien ce projet, plusieurs outils technologiques ont été utilisés. Voici les principaux :

- **Logiciel de design:** Lunacy
- **Logiciel de modelisation:** draw.io
- **IDE**: Visual Studio Code est l'editeur de choix pour notre project.
- **Syteme de controle de version**: Git et Github.
- **Langages de Programmation et technologie**: JavaScript pour front-end(Vue.JS) et FireBase en back-end.

2.2 Méthode utilisé

Comme méthode utiliser, nous avont opter pour la méthode de dévélopement agile SCRUM et la modelisation UML. Chaque tâches sont séparé en sprint, et à la fin de chacun d'entre eux, un livrable est fournie. La liste des livrables et de l'application elle meme attendues pour ce project sont:

- L'Application de gestion de tâche GETDO.
- Creation de tâche.
- Vision Board.
- Tableaux de bord et visualisations de données.
- Prise en charge d'un styles de gestion des tâches.

2.2.1 Mise en place des outils de gestion de project

Pour assuré la gestion bonne gestion du project, nous allons utiliser le logiciel de gestion JIRA

2.2.2 Conseption du project

Diagramme de cas d'utilisation et description textuelle

Dans l'optique d'offire une experience utilisateur cohérante, nous nous devons de representer les diverses intéractions que l'utilisateur aura avec la plate forme. C'est pour cela que nous avons opter pour le diagramme de cas d'utilisation suivant:

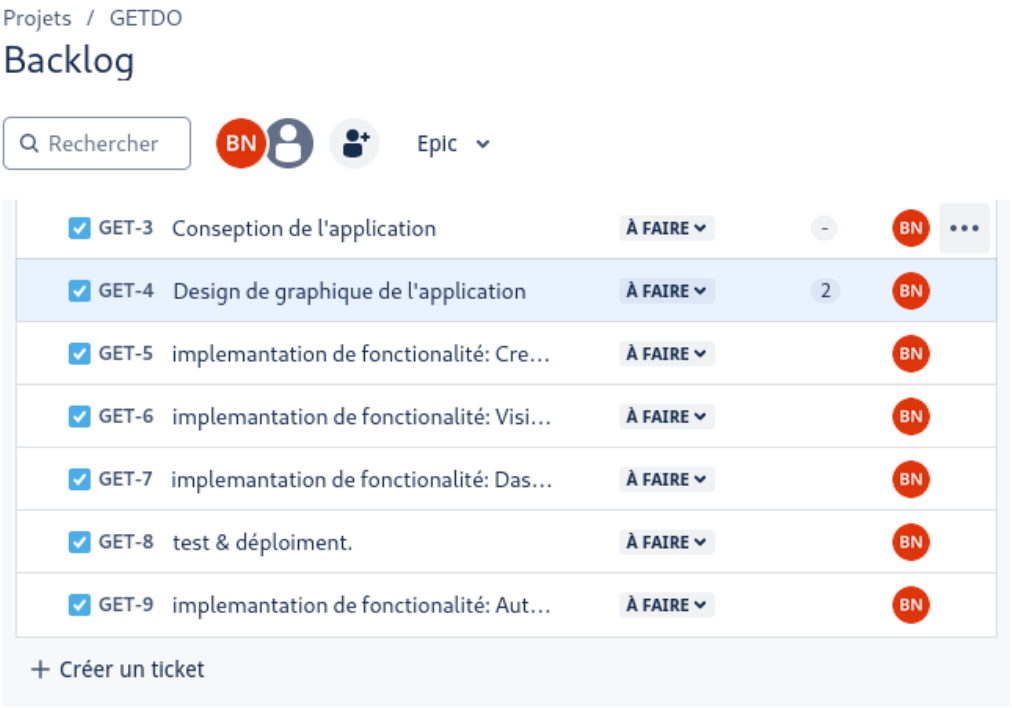

Figure 2.1: Vue d'ensemble d'une tâche dans un projet Jira

Système: Système de gestion de tâches

Acteurs:

- Utilisateur: Utilise le système pour créer, gérer et suivre ses tâches.
- Administrateur: Gère les utilisateurs, les paramètres du système et les rapports.

Cas d'utilisation:

- Connexion/Inscription: L'utilisateur se connecte ou crée un compte.

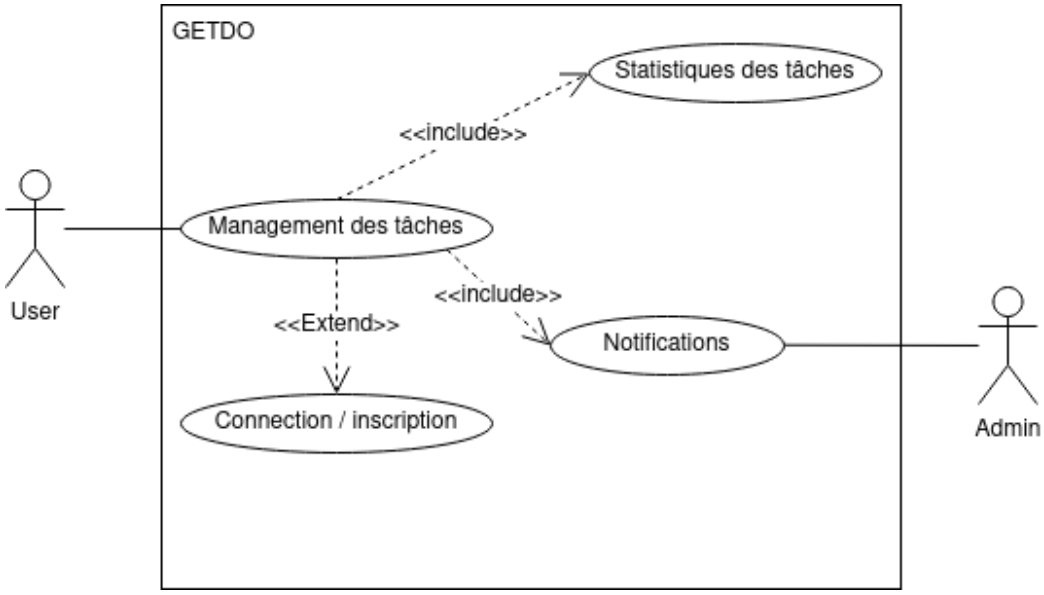

Figure 2.2: Diagramme de cas d'utilisation de GETDO

- Gestion des tâches: L'utilisateur crée, modifie, supprime et organise ses tâches.
- Statistiques des tâches: L'utilisateur génère des rapports (tâches terminées, temps passé, tâches en retard).
- Notifications: L'utilisateur consulte les notifications envoyer par l'administrateur

Relations:

- Utilisateur: Peut utiliser tous les cas d'utilisation.
- Administrateur: Peut envoyer des notifications a l'utilisateur.
- Gestion des tâches: Inclut les notifications.
- Statistiques des tâches: Étend la gestion des tâches.

Senario de cas d'utilisation

Scénario utilisateur: Création d'une tâche

Acteur: Utilisateur

Cas d'utilisation: Gestion des tâches

Objectif: Créer une nouvelle tâche dans le système.

Étapes:

- L'utilisateur se connecte au système en utilisant son nom d'utilisateur et son mot de passe.
- L'utilisateur sélectionne l'option "Créer une tâche" dans le menu principal.
- L'utilisateur entre le titre de la tâche, une description détaillée, la date d'échéance et la priorité de la tâche.
- L'utilisateur clique sur le bouton "Enregistrer" pour créer la tâche.
- Le système enregistre la tâche et affiche un message de confirmation.
- L'utilisateur peut ensuite visualiser la tâche dans sa liste de tâches ou dans son calendrier.

Variations: L'utilisateur peut définir une date de rappel pour la tâche. **Résultat:** Une nouvelle tâche est créée dans le système et est visible pour l'utilisateur qui l'a créée et pour les utilisatePlaurs auxquels elle a été attribuée.

Scénario utilisateur: Modification d'une tâche

Acteur: Utilisateur

Cas d'utilisation: Gestion des tâches

Objectif: Modifier une tâche existante dans le système.

Étapes:

- L'utilisateur se connecte au système en utilisant son nom d'utilisateur et son mot de passe.
- L'utilisateur sélectionne la tâche qu'il souhaite modifier dans sa liste de tâches ou dans son calendrier.
- L'utilisateur modifie les informations de la tâche, telles que le titre, la description, la date d'échéance, la priorité, les étiquettes, les pièces jointes et les commentaires.
- L'utilisateur clique sur le bouton "Enregistrer" pour enregistrer les modifications.
- Le système enregistre les modifications et affiche un message de confirmation.
- Les modifications de la tâche sont reflétées dans la liste des tâches de l'utilisateur et dans le calendrier.

Variations:

- L'utilisateur peut modifier l'état de la tâche, par exemple de "En cours" à "Terminée".
- L'utilisateur peut modifier la date de rappel de la tâche.

Résultat: La tâche est modifiée dans le système et les modifications sont reflétées pour tous les utilisateurs qui peuvent voir la tâche.

Scénario utilisateur: Suppression d'une tâche

Acteur: Utilisateur

Cas d'utilisation: Gestion des tâches

Objectif: Supprimer une tâche existante du système.

Étapes:

- L'utilisateur se connecte au système en utilisant son nom d'utilisateur et son mot de passe.
- L'utilisateur sélectionne la tâche qu'il souhaite supprimer dans sa liste de tâches ou dans son calendrier.
- L'utilisateur clique sur le bouton "Supprimer" pour supprimer la tâche.
- Le système affiche une confirmation demandant à l'utilisateur de confirmer la suppression.
- L'utilisateur clique sur le bouton "Confirmer" pour supprimer définitivement la tâche.
- Le système supprime la tâche et affiche un message de confirmation.

Résultat: La tâche est supprimée du système et n'est plus visible pour les utilisateurs.

2.2.3 Architecture de l'application

L'architecture d'une application est cruciale pour assurer sa performance, sa maintenabilité et son évolutivité. Dans le cadre de notre projet, nous avons choisi d'utiliser **Vue.js** comme framework frontend et **Firebase** comme solution backend. Cette combinaison, offre une approche modulaire et efficace permettant de répondre rapidement aux besoins des utilisateurs tout en garantissant une intégration fluide des différentes fonctionnalités. Dans cette, section nous allons détailler les composants clés de l'architecture de

l'application ainsi que les interactions entre le frontend et le backend, afin de mieux comprendre les choix techniques effectués et leur impact sur le développement global du.

frontend

Le frontend d'une application constitue l'interface, utilisateur celle par laquelle les utilisateurs interagissent directement avec les fonctionnalités. proposées Dans notre, projet nous avons opté pour.Vue, js un framework JavaScript reconnu pour sa flexibilité et sa capacité à créer des interfaces dynamiques et. réactives Cette section explore les principaux composants du, frontend y compris la structure des composants.

Liste des vues:

- **vue d accueil:** LandingVue
- **Vue'd inscription:** InscriptionVue
- **Vue de connection:** ConnectionVue
- **vue des tâches:** HomeVue
- **vue du vision bord:** VisionBordVue
- **vue du calendrier:** CalendarVue
- **vue du dashbord:** DashbordVue

Liste des composants:

- HeaderComponent
- TaskComponent
- VisionComponent

Structure La structure du frontend de l'application:

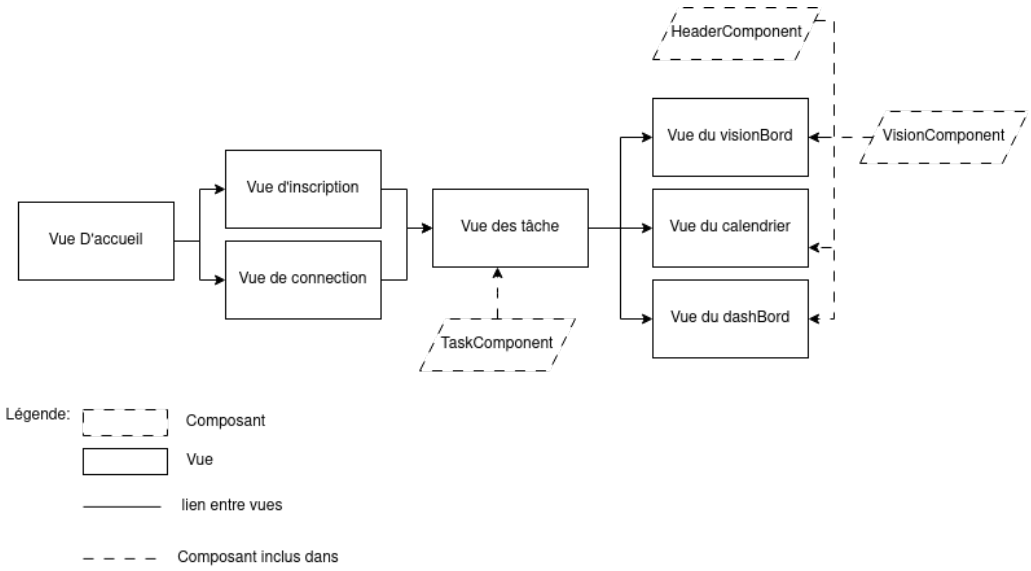

Figure 2.3: Structure des vues de'l

Backend

Le backend d'une application est essentiel pour gérer les données, l'authentification et la logique métier qui soutiennent le fonctionnement de l'interface utilisateur. Dans notre projet, nous avons choisi Firebase comme solution backend, qui offre une gamme complète de services intégrés pour simplifier le développement. Nous nous pancheront sur les principaux composants du backend, y compris l'authentification des utilisateurs et les bases de données en expliquant comment ils interagissent pour garantir la sécurité, la scalabilité et la performance de l'application. Grâce à Firebase, nous avons pu tirer parti d'une architecture serverless, permettant ainsi une intégration efficace et rapide des fonctionnalités nécessaires à notre projet.

Présentation Générale de Firebase

Figure 2.4: firebase

Firebase est une plateforme complète qui fournit tout ce dont vous avez besoin pour créer, développer et gérer des applications mobiles et web performantes. Avec ses outils puissants et faciles à utiliser, Firebase permet aux développeurs de gagner du temps et de se concentrer sur la création de fonctionnalités innovantes plutôt que sur la gestion de l'infrastructure sous-jacente. Firebase, dispose des fonctionalité suivante:

- Authentification : Gérer les connexions des utilisateurs et fournir un accès sécurisé aux données
- Bases de données : Stocker et gérer des données structurées et non structurées
- Stockage : Stocker et gérer des fichiers tels que des images, des vidéos et des documents
- Fonctions cloud : Exécuter du code sans serveur en réponse à des événements
- Notifications push : Envoyer des notifications aux utilisateurs sur leurs appareils

- Analyse : Suivre les performances de l'application et le comportement des utilisateurs

Interaction frontend-backend

L'interaction entre le frontend et le backend est cruciale pour les applications web modernes. Dans notre projet, nous utilisons Vue.js pour le frontend et Firebase pour le backend. Vue.js permet de créer des interfaces utilisateur interactives et performantes, tandis que Firebase offre une solution complète incluant l'authentification, la gestion de bases de données en temps réel, et le stockage. Les details d'intéraction se feront comme illustré si desous:

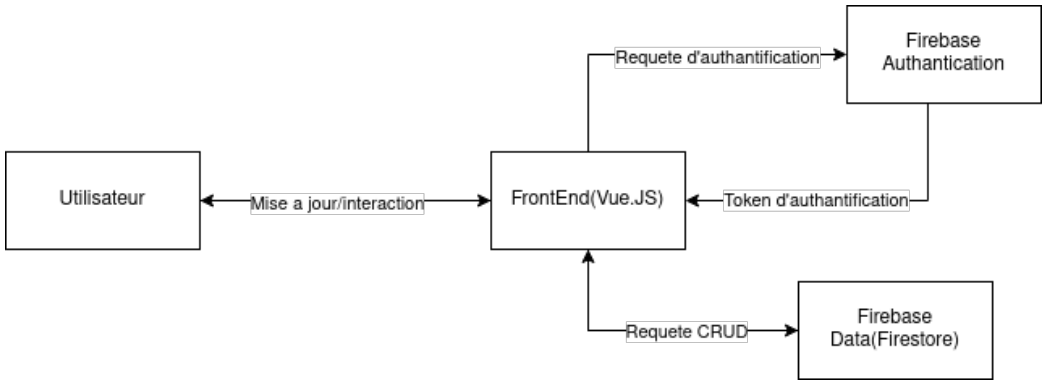

Figure 2.5: Interaction frontend-backend

- **Utilisateur à Vue.js** : L'utilisateur interagit avec l'interface Vue.js.
- **Vue.js à Firebase Authentication** : Vue.js envoie des requêtes d'authentification pour vérifier l'utilisateur.
- **Firebase Authentication à Vue.js** : Firebase Authentication renvoie un token d'authentification.
- **Vue.js à Firebase Database** : Vue.js envoie des requêtes CRUD (Create, Read, Update, Delete) à la base de données Firebase.

- **Firebase Database à Vue.js** : Firebase Database renvoie les réponses des opérations CRUD.
- **Vue.js à Utilisateur** : Vue.js met à jour l'interface utilisateur avec les données reçues.

2.2.4 Identité visuelle du project

L'identité visuelle est l'élément crucial qui donne vie à la personnalité et aux valeurs d'un projet. Elle constitue la première impression que les utilisateurs auront de votre application et joue un rôle essentiel dans la création d'une expérience utilisateur cohérente et engageante.

Cette section du plan de projet s'attache à définir les principes fondamentaux qui guideront la création de l'identité visuelle de l'application. Elle couvrira les aspects suivants :

- **Logo et charte graphique:** Définir les éléments visuels clés qui représentent l'identité de l'application, tels que le logo, la palette de couleurs, la typographie et les images.
- **Directives de style:** Établir des règles claires et cohérentes pour l'utilisation de l'identité visuelle dans tous les supports de communication, y compris l'interface utilisateur, les documents marketing et les supports promotionnels.

Logo et charte graphique

Le logo et la charte graphique sont les piliers fondamentaux de l'identité visuelle d'un projet. Ils constituent la représentation visuelle de la marque et jouent un rôle crucial dans la communication des valeurs, de la mission et

de la personnalité du projet auprès du public cible.

GETDO

Figure 2.6: Logo de GETDO

Directive de style

Dans le cadre de l'implémentation de notre application web, il est essentiel d'adopter une approche cohérente et structurée. Cette section présente les directives de style que nous avons suivies tout au long du projet. Ces règles visent à assurer une expérience utilisateur optimale. En suivant ces directives, nous avons pu créer, répondant aux besoins des utilisateurs tout en respectant les meilleures pratiques de développement web.

- **Palette de couleur principale**:

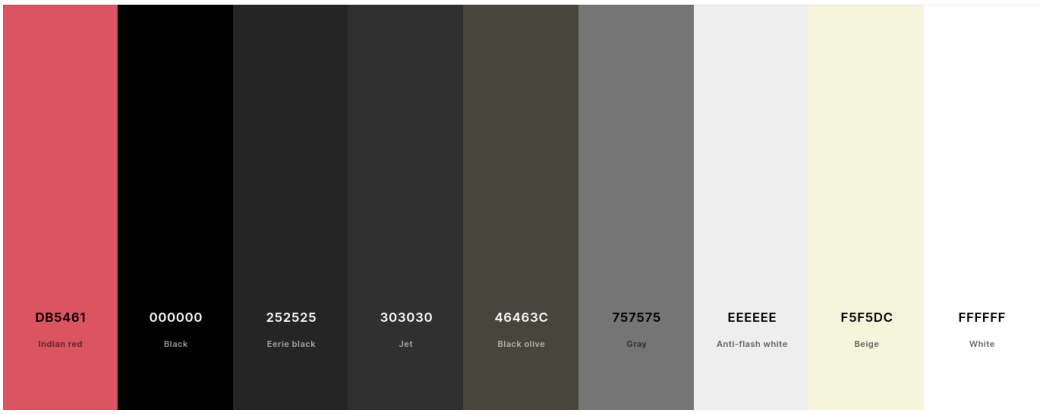

Figure 2.7: Palette de couleur principale

De gauche a droite nous avons:

- **Indian red(DB5461)**: Uiliser comme couleur principale de l'application
- **Black(000000)**:Couleur des zone de texte en mode sombre et du texte en mode claire.
- **Eerie black(252525)**:Couleur des espaces dans les boites de dialogue.
- **gauche a droite nous avons:Jet(303030)**:Couleur des boites de dialogue (en mode sombre)
- **Black olive(46463C)**:Couleur d'arrière plan (en mode sombre)
- **Grey(757575)**:Couleur des liens
- **Anti-flash white(EEEEEE)**:Couleur des boites de dialogue (en mode clair)
- **Beige(F5F5DC)**:Couleur d'arrière plan (en mode claire)
- **White(FFFFFF)**:Couleur des zone de texte en mode claire et

du texte en mode sombre.

- **palette de couleur secondaire**: Celle ci sera utiliser pour les indicateurs de niveaux d'importance des tâches

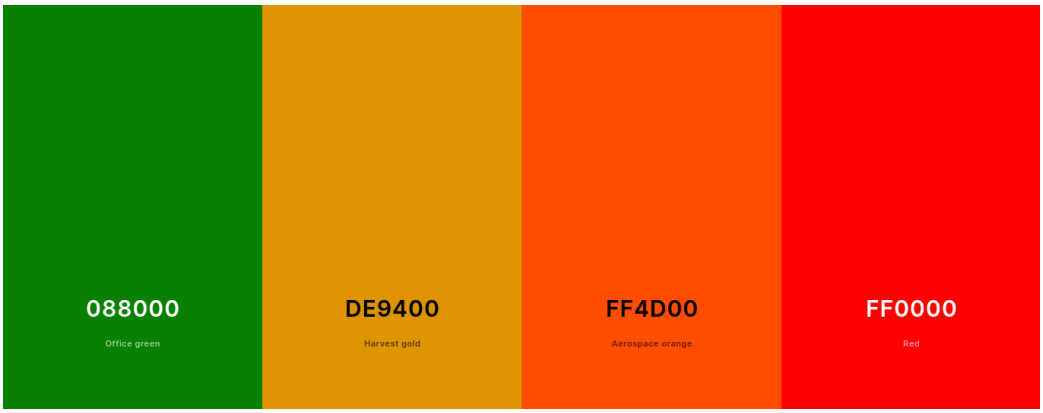

Figure 2.8: Palette de couleur secondaire

De gauche a droite nous avons:

- **Office green(088000)**: Pour les tâches de moindre importance.
- **Harvest gold(DE9400)**: Pour les tâches d'importance faible.
- **Aerospace orange(FF4D00)**: Pour les tâches importante.
- **Red(FF0000)**: Pour les tâches urgente.

Chapitre 3

Analyse des resultats

Dans ce chapitre, nous nous pencherons sur l'analyse des résultats obtenus suite à l'implémentation de notre application de gestion de tâches. Cette phase d'analyse est cruciale pour évaluer l'efficacité, la performance et la convivialité de l'application, ainsi que pour identifier les éventuels axes d'amélioration. Nous commencerons par examiner les objectifs initiaux du projet et comparerons ces objectifs avec les résultats atteints. Cela nous permettra de déterminer dans quelle mesure les attentes ont été satisfaites et d'évaluer l'impact réel de l'application sur la gestion des tâches. Enfin, nous explorerons les avantages et les limitations de l'utilisation de Vue.js et Firebase dans le contexte de ce projet. Cette section fournira un retour d'expérience précieux sur le choix de ces technologies pour le développement de l'application.

3.1 Évaluation des Objectifs Initiaux

3.1.1 Objectifs du Projet

L'objectif principal de notre projet était de développer une application de gestion de tâches, nommée GETDO, qui offre une expérience utilisateur intuitive et efficace pour la gestion quotidienne des tâches. Pour atteindre cet objectif, nous avons défini plusieurs fonctionnalités clés que l'application devait intégrer.

- **Création de Tâche** L'une des fonctionnalités fondamentales de GETDO est la création de tâches. L'application devait permettre aux utilisateurs de créer des tâches rapidement et facilement, en fournissant des options pour définir des titres, des descriptions, des dates d'échéance, des priorités et des catégories. Cette fonctionnalité vise à aider les utilisateurs à organiser et prioriser leurs tâches de manière claire et structurée.

- **Vision Board** Le Vision Board est une fonctionnalité innovante de GETDO qui permet aux utilisateurs de visualiser leurs tâches sous forme de tableaux interactifs. Cette vue visuelle aide les utilisateurs à obtenir une perspective globale de leurs tâches, à organiser leurs projets de manière visuelle et à suivre l'avancement de leurs objectifs à long terme. Le Vision Board devait offrir une interface glisser-déposer intuitive pour faciliter la gestion des tâches.

- **Tableaux de Bord et Visualisations de Données** Pour fournir une vue d'ensemble complète et analytique de la gestion des tâches, GETDO devait intégrer des tableaux de bord et des visualisations de données. Cette fonctionnalité permet aux utilisateurs de visualiser

leurs performances, de suivre le progrès de leurs tâches et de générer des rapports. Les tableaux de bord devaient inclure des graphiques et des statistiques sur les tâches accomplies, en cours et en retard, offrant ainsi une vision claire de la productivité des utilisateurs.

- **Prise en Charge de Styles de Gestion des Tâches** GETDO devait offrir une flexibilité dans la gestion des tâches en prenant en charge différents styles de gestion. Les utilisateurs pouvaient choisir entre plusieurs méthodes de gestion des tâches, telles que le Kanban, la méthode GTD (Getting Things Done) ou des listes de tâches traditionnelles. Cette flexibilité permet aux utilisateurs d'adopter le style de gestion qui correspond le mieux à leurs besoins et préférences.

Ces fonctionnalités clés définissaient les objectifs initiaux de notre projet, visant à créer une application de gestion de tâches complète, flexible et adaptée à divers styles de gestion. En atteignant ces objectifs, nous souhaitions offrir une solution qui améliore la productivité et l'organisation des utilisateurs dans leur vie quotidienne et professionnelle.

3.1.2 Résultats Atteints

Cette sous-section présente une évaluation des résultats obtenus par rapport aux objectifs initiaux du projet. Nous analyserons les fonctionnalités implémentées et comparerons les résultats atteints avec les attentes définies. Cela nous permettra de mesurer le succès de notre application de gestion de tâches GETDO et de mettre en lumière les points forts ainsi que les aspects nécessitant des améliorations.

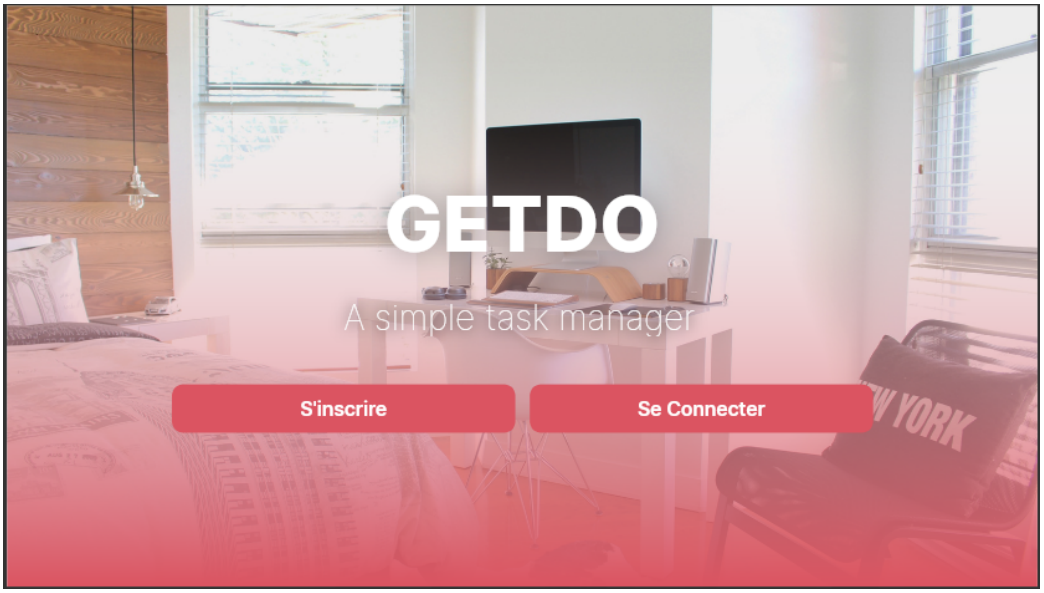

Figure 3.1: Page d'accueil

Figure 3.2: Page d'inscription

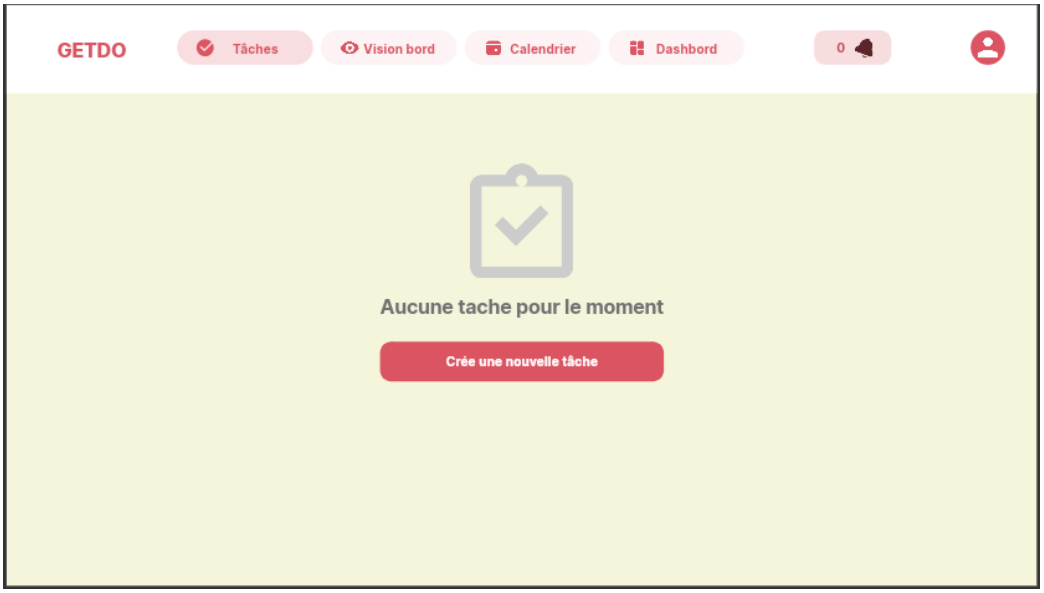

Figure 3.3: Page de visualisation de tâche(vide)

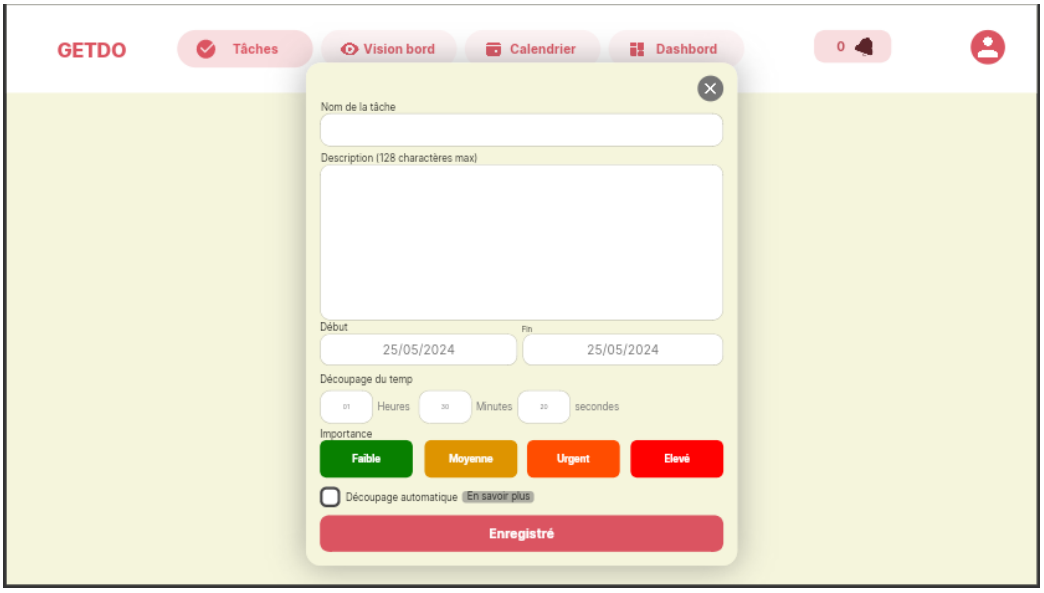

Figure 3.4: Page de visualisation de tâche(formulaire de creation)

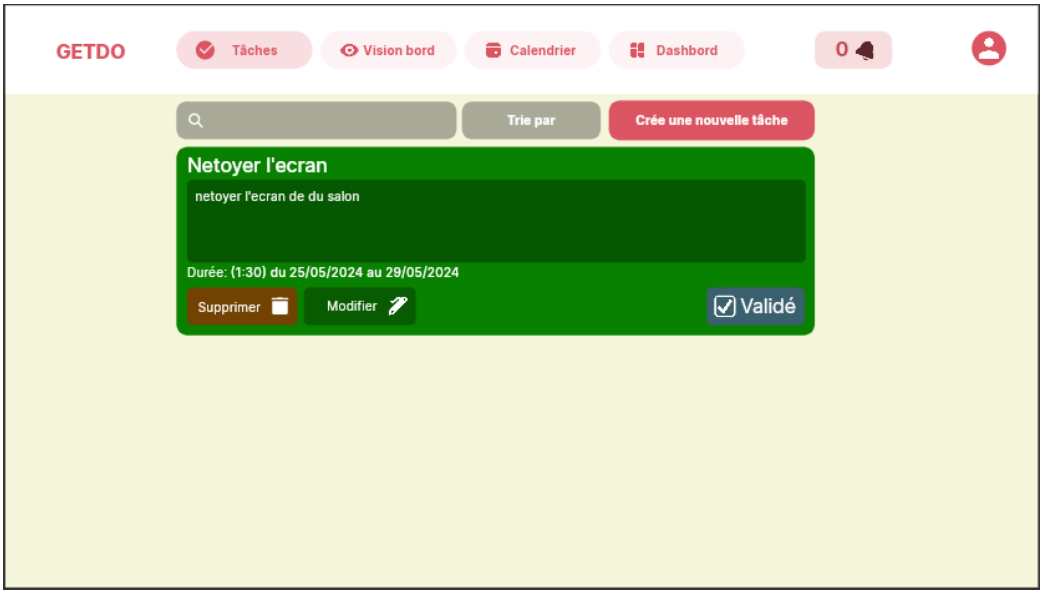

Figure 3.5: Page de visualisation de tâche apres creation d'une tâche

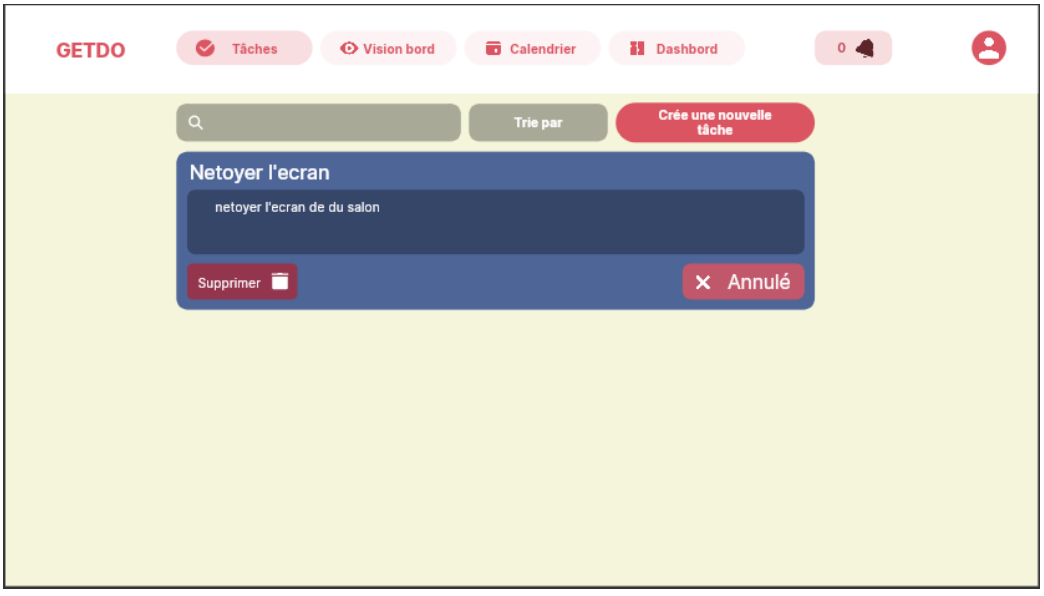

Figure 3.6: Tâche apres qu'elle soit terminer

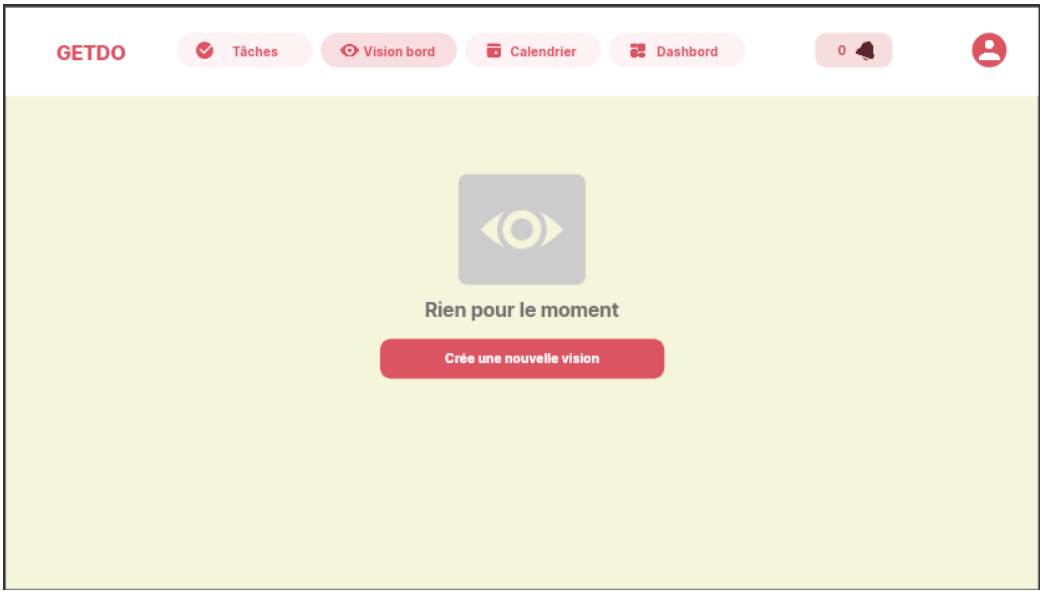

Figure 3.7: Vision bord(vide)

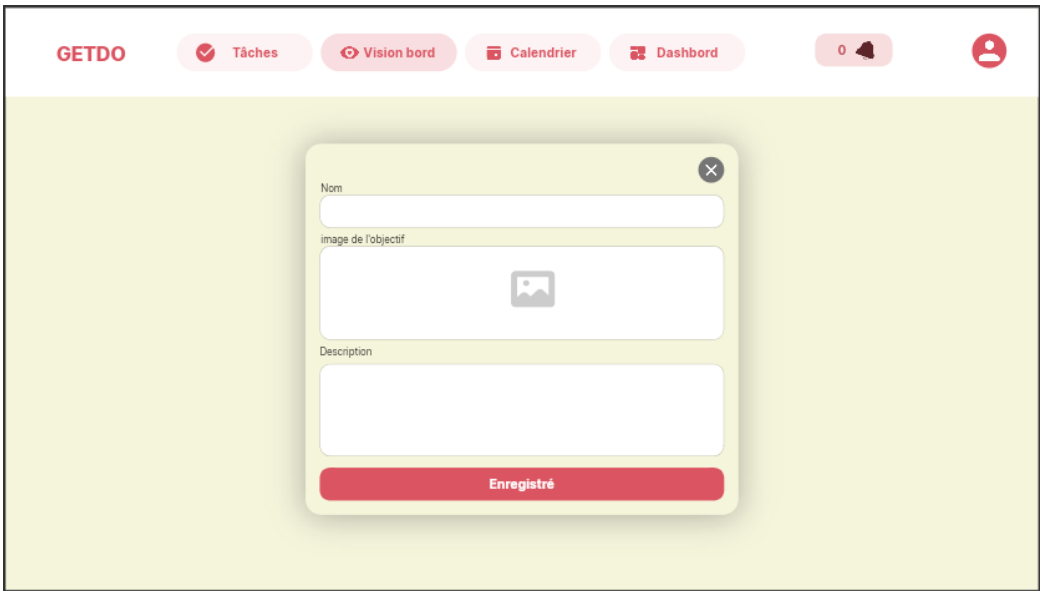

Figure 3.8: Vision bord(formulaire de creation)

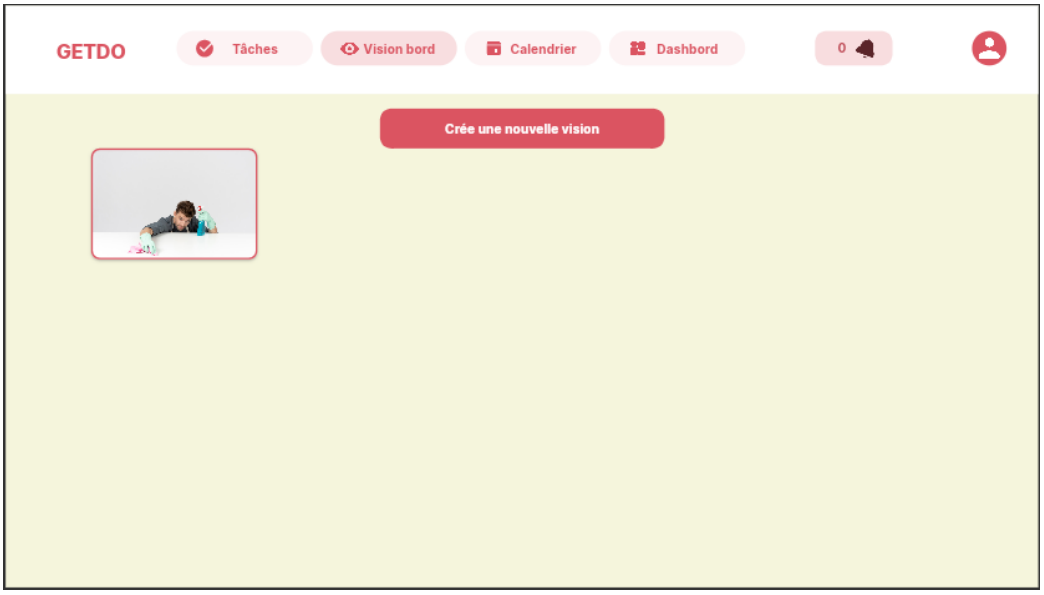

Figure 3.9: Vision bord(photo ajouter)

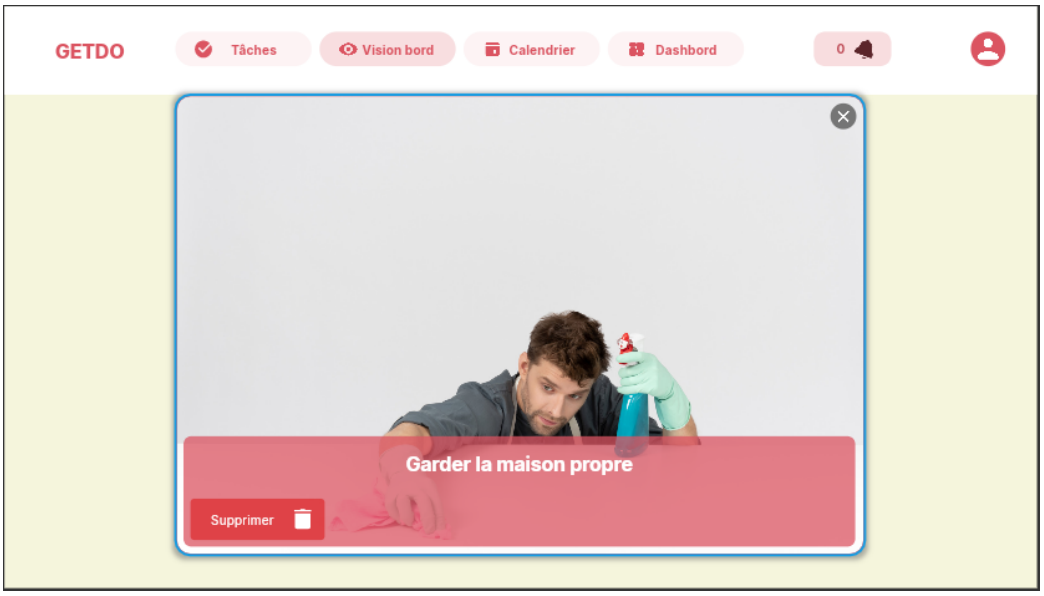

Figure 3.10: Vue de plus raproché de la photo et objectif associer

3.2 Avantages et Limitations des Technologies Utilisées

Dans cette sous-section, nous analyserons les avantages et les limitations des technologies choisies pour le développement de notre application de gestion de tâches, GETDO. L'utilisation de Vue.js pour le frontend et de Firebase pour le backend a offert de nombreux bénéfices, mais aussi posé certains défis. Nous évaluerons comment ces technologies ont contribué à la réalisation de nos objectifs et examinerons les contraintes rencontrées lors de leur mise en uvre

3.2.1 Avantages de Vue.js

Vue.js a été choisi pour le développement du frontend de notre application GETDO en raison de ses nombreux avantages. Voici les principaux bénéfices observés lors de son utilisation :

- **Facilité d'Apprentissage et d'Utilisation:**Vue.js est réputé pour sa courbe d'apprentissage douce, ce qui a permis à notre équipe de développement de se familiariser rapidement avec le framework. Sa documentation claire et exhaustive a également facilité l'intégration des nouvelles fonctionnalités et la résolution des problèmes rencontrés.
- **Performances et Optimisation:**Vue.js est optimisé pour offrir de bonnes performances grâce à son Virtual DOM et à son mécanisme de réactivité. Cela a permis à l'application GETDO de fonctionner de manière fluide, même avec un grand nombre de tâches à gérer, garantissant une expérience utilisateur agréable et sans ralentissement.
- **Support Actif et Communauté Vibrante:** Vue.js bénéficie d'une

communauté active et d'un support continu de la part de ses développeurs. Cette communauté dynamique a fourni de nombreuses ressources, plugins et solutions aux défis rencontrés, accélérant ainsi le développement et la résolution des problèmes.

- **Architecture Modulaire et Réactive:** Vue.js offre une architecture réactive qui permet de créer des interfaces utilisateur dynamiques et interactives. Sa modularité a facilité la séparation des préoccupations, permettant de structurer l'application en composants réutilisables et maintenables. Cette approche a simplifié le développement et la maintenance de l'application.
- **Intégration Facile avec d'Autres Bibliothèques:** L'écosystème de Vue.js est conçu pour s'intégrer facilement avec d'autres bibliothèques et outils, tels que Vuex pour la gestion de l'état et Vue Router pour la navigation. Cette flexibilité a permis une adaptation facile aux besoins spécifiques du projet et une extension des fonctionnalités de l'application sans complexité excessive.

3.2.2 Avantages de Firebase

Firebase, choisi comme backend pour le développement de l'application GETDO, offre une gamme complète de services intégrés qui ont grandement facilité la mise en uvre du projet. Voici les principaux avantages de l'utilisation de Firebase :

- **Solution Tout-en-Un:** Firebase fournit une suite complète de services backend, incluant la base de données en temps réel (Firestore), l'authentification des utilisateurs, le stockage de fichiers, les fonctions cloud, et bien plus. Cette solution tout-en-un a permis de centraliser les

besoins backend en un seul service, simplifiant ainsi le développement et la gestion de l'application.

- **Authentification Facile et Sécurisée:** Firebase Authentication offre des options simples et sécurisées pour la gestion des utilisateurs, incluant l'authentification par email/mot de passe, l'authentification via des fournisseurs tiers (Google, Facebook, etc.), et la gestion des sessions utilisateur. Cela a permis d'implémenter rapidement des fonctionnalités d'authentification robustes et sécurisées.
- **Fonctionnalités Serverless:** Avec Firebase Cloud Functions, il est possible de déployer des fonctions serverless qui réagissent aux événements de la base de données, aux requêtes HTTP, et plus encore. Cette capacité a permis d'implémenter des logiques métiers complexes de manière flexible et évolutive sans avoir à gérer des serveurs dédiés.
- **Facilité d'Intégration et Documentation Complète:** Firebase dispose d'une documentation exhaustive et d'un support de qualité, facilitant l'intégration rapide des services dans l'application. Cette documentation a été particulièrement utile pour résoudre les problèmes techniques et implémenter les fonctionnalités avancées de manière efficace.

3.2.3 Limitations de Vue.js et Firebase

Limitations de Vue.js

Bien que Vue.js présente de nombreux avantages, certaines limitations ont été rencontrées au cours du développement de l'application GETDO : Courbe d'Apprentissage pour les Concepts Avancés

- **Complexité Croissante :** Bien que Vue.js soit facile à apprendre pour les débutants, la maîtrise des concepts avancés comme Vuex (gestion d'état), les mixins, et les plugins peut être complexe et nécessiter un temps d'adaptation significatif.
- **Fragmentation des Plugins :** L'écosystème de Vue.js évolue rapidement, ce qui peut entraîner des problèmes de compatibilité entre différentes versions des plugins et des bibliothèques. Cela nécessite une veille technologique constante et peut rendre la maintenance du projet plus difficile.
- **Ralentissement :** Lorsqu'il s'agit de gérer des listes très longues ou des tableaux de données volumineux, Vue.js peut rencontrer des problèmes de performance, malgré l'optimisation du Virtual DOM. Des techniques avancées et des optimisations supplémentaires peuvent être nécessaires pour maintenir une interface fluide.

Limitations de Firebase

Firebase, malgré ses nombreux atouts, présente également certaines limitations qui ont impacté le développement et l'utilisation de l'application GETDO :

- **Lock-in Technologique :** Firebase est une plateforme propriétaire de Google, ce qui signifie que les développeurs sont en grande partie liés à l'écosystème de Google. Migrer vers une autre solution backend peut être complexe et coûteux en termes de temps et de ressources.
- **Modèle de Tarification :** Bien que Firebase propose une tarification gratuite pour les petits projets, les coûts peuvent rapidement augmenter avec l'échelle et l'usage intensif des services. Les fonctionnalités

avancées et les requêtes fréquentes peuvent entraîner des frais importants, surtout pour les startups ou les projets à budget limité.

- **Complexité des Requêtes :** Firestore, bien que puissant, peut rencontrer des limitations lors de la réalisation de requêtes complexes. Les développeurs doivent souvent trouver des solutions de contournement pour des opérations qui seraient triviales avec des bases de données relationnelles traditionnelles.
- **Connexion en Temps Réel :** Firebase dépend d'une connexion Internet stable pour synchroniser les données en temps réel. Dans des environnements où la connectivité est limitée ou instable, cela peut poser des problèmes de synchronisation et d'accès aux données.
- **Règles de Sécurité :** La configuration des règles de sécurité pour Firestore peut être complexe et sujette à des erreurs. Une mauvaise configuration peut entraîner des failles de sécurité potentielles, exposant les données des utilisateurs à des risques.

3.3 Perspective d'amelioration

Retour des Utilisateurs

- **Feedback Utilisateur Continu :**Mettre en place des mécanismes pour recueillir régulièrement les retours des utilisateurs afin d'identifier les besoins émergents et les points de friction.
- **Mise en uvre de Nouvelles Fonctionnalités :** Intégrer de nouvelles fonctionnalités basées sur les retours des utilisateurs, telles que des rappels par notification, des intégrations avec d'autres outils de productivité, ou des options de personnalisation avancées.

- **Intégration du dashbord**: Lors de prochaine mise a jour nous incluront un tableau de visualisation des données

Amélioration de l'Expérience Utilisateur

- **Refonte de l'interface:** Améliorer l'interface utilisateur pour la rendre plus intuitive et accessible, en tenant compte des meilleures pratiques de design UX.
- **Accessibilité :** S'assurer que l'application est accessible à tous les utilisateurs, y compris ceux ayant des besoins spécifiques en matière d'accessibilité.

Technologies et Outils

- **Alternatives Backend :** Évaluer d'autres solutions backend pour compléter ou remplacer certaines fonctionnalités de Firebase, en cas de besoin spécifique ou de contraintes de coût.
- **Mises à Jour et Nouveautés :** Rester à jour avec les dernières versions de Vue.js et Firebase, ainsi que les nouvelles fonctionnalités et améliorations proposées par la communauté et les développeurs de ces technologies.

Intégration de l'inteligence Artificielle

Grace a l'intégration de l'inteligence Artificielle, notre service aura la possibilité de simplifier la creation et gestion de tâche.

En mettant en uvre ces perspectives d'amélioration, nous visons à continuer à offrir une application de gestion de tâches performante, sécurisée et centrée sur l'utilisateur. Ces efforts contribueront à maintenir la satisfaction

des utilisateurs et à assurer la croissance et l'évolution de GETDO dans un environnement technologique en constante évolution.

Conclusion Générale

En conclusion, ce rapport a détaillé lensemble du processus de développement de lapplication de gestion de tâches GETDO, allant de la conception initiale à lévaluation des résultats obtenus. Lobjectif principal de ce projet était de créer une solution robuste et intuitive pour aider les utilisateurs à gérer leurs tâches personnelles et professionnelles de manière efficace.

Nous avons abordé les différentes phases de développement, en mettant en évidence les défis rencontrés et les solutions apportées. Le choix des technologies modernes telles que Vue.js pour le front-end et Firebase pour le back-end a permis de réaliser une application performante, réactive et sécurisée.

Lanalyse des résultats a démontré que les objectifs initiaux du projet ont été atteints, voire dépassés dans certains aspects. Lapplication GETDO a prouvé son efficacité en termes de gestion de tâches, offrant une interface utilisateur conviviale et des fonctionnalités avancées telles que le Vision Board, les tableaux de bord et les visualisations de données.

Cependant, nous avons également identifié des limitations et des axes damélioration, notamment en ce qui concerne la scalabilité et la gestion des tâches complexes. Ces observations ouvrent des perspectives intéressantes pour des évolutions futures, notamment lintégration de lintelligence artificielle pour

améliorer encore davantage lexpérience utilisateur et la productivité.

Enfin, ce projet a été une occasion précieuse dacquérir des compétences pratiques en développement dapplications web et dexplorer les possibilités offertes par des technologies modernes. Nous espérons que ce travail pourra servir de base pour des projets futurs et contribuer à lamélioration continue des outils de gestion de tâches.

Reference bibiographique

- https://blog.visual-paradigm.com/fr/
- https://firebase.google.com/
- https://coolors.co/
- https://app.diagrams.net/
- https://vuefire.vuejs.org/
- https://www.overleaf.com/
- https://www.latex-fr.net/
- https://www.digitalocean.com/