



Licence Professionnelle en Génie Logiciel

RAPPORT FINAL DU PROJET TUTEURE 2

Liste des étudiants participants :

- KANQUAYE Kankolé Gauthier O.D.
- KOUDOZIAN Kossi Senyo
- SAÏBOU Chouaïbu

UE : INF1525 – Projet Tuteuré GL 2

Tuteur : M. AKAKPO

Année académique : 2025-2026

Table des matières

1.	Informations générales	2
2.	Introduction	2
3.	Objectifs du projet	2
4.	Technologies utilisées	2
4.1	Front-End.....	2
4.2	Back-End / Base de données	3
4.3	Autres éléments	3
5.	Fonctionnalités du projet et état d'avancement.....	3
5.1	Tableau de bord (Dashboard).....	3
5.2	Transactions	3
5.3	Comptes	3
5.4	Enveloppes budgétaires (Budget)	3
5.5	Épargne	4
5.6	Abonnements.....	4
5.7	Rapports.....	4
5.8	ChatBot (Assistant).....	5
5.9	Paramètres	5
6.	Navigation principale	5
7.	Base de données (Firebase)	6
7.1	Firebase Authentication	6
7.2	Cloud Firestore	6
8.	Phase actuelle : Tests & validation	6
8.1	Objectif de la phase de test.....	6
9.	Livrables attendus et situation actuelle	8
9.1	Liste des livrables	8
10.	Calendrier de réalisation (planning)	7
11.	Bases de données	8
12.	Guide de déploiement.....	10
13.	Guide d'utilisation	13
14.	Difficultés rencontrées	27
15.	Conclusion	28

1. Informations générales

- Nom du projet : Miabe Budget Manager
- Type : Application Web de gestion financière personnelle
- Langages : HTML, CSS, JavaScript
- Base de données : Firebase (Cloud Firestore)
- Authentification : Firebase Authentication
- Phase actuelle : Tests & validation finale
- Tuteur : M. AKAKPO

2. Introduction

Le projet Miabe Budget Manager est une application web conçue pour aider les utilisateurs à gérer efficacement leurs finances personnelles.

Elle permet de suivre les transactions, gérer plusieurs comptes, créer des enveloppes budgétaires, suivre l'épargne, analyser les abonnements, consulter des rapports et recevoir des conseils via un chatbot intégré.

L'objectif principal est de fournir une solution simple, moderne et complète, accessible à tout utilisateur souhaitant organiser son budget et mieux contrôler ses dépenses.

3. Objectifs du projet

Les objectifs du projet sont :

- Permettre à l'utilisateur de suivre ses revenus et dépenses
- Offrir une vision claire via un tableau de bord avec graphiques
- Permettre la gestion de comptes financiers multiples
- Mettre en place un système de budget par enveloppes
- Gérer l'épargne libre et les objectifs d'épargne
- Suivre les abonnements et détecter les dépenses inutiles
- Générer des rapports financiers détaillés
- Fournir un assistant (chatbot) pour analyser les données et donner des conseils
- Proposer une interface moderne avec paramètres et sécurité

4. Technologies utilisées

4.1 Front-End

- HTML5
- CSS3

- JavaScript

4.2 Back-End / Base de données

- Firebase Authentication (gestion utilisateurs)
- Firebase Cloud Firestore (stockage des données)

4.3 Autres éléments

- Graphiques (statistiques et visualisations)
- Export des transactions (CSV / PDF / Excel)
- Système de notifications internes

5. Fonctionnalités du projet et état d'avancement

À l'heure actuelle, toutes les fonctionnalités prévues sont implémentées. Le projet est en phase de tests et correction de bugs avant livraison finale.

5.1 Tableau de bord (Dashboard)

Fonctionnalités :

- Affichage du solde total
- Affichage des comptes sous forme de cartes
- Graphique évolution des revenus
- Graphique évolution des dépenses
- Affichage des dernières transactions (5 à 10)

5.2 Transactions

Fonctionnalités :

- Liste complète des transactions
- Ajouter / Modifier / Supprimer une transaction
- Recherche texte
- Filtre par compte
- Filtre par période (date début / date fin)
- Export CSV / PDF / Excel

5.3 Comptes

Fonctionnalités :

- Ajouter un compte
- Supprimer un compte
- Affichage sous forme de cartes
- Sécurité : authentification requise (PIN / mot de passe / empreinte)

5.4 Enveloppes budgétaires (Budget)

Fonctionnalités :

- Ajouter une enveloppe

- Supprimer une enveloppe
- Limite mensuelle par catégorie (ex : alimentation)
- Calcul automatique :
 - montant dépensé
 - montant restant
 - barre de progression
- Notification enveloppe pleine

5.5 Épargne

A. Épargne libre

Fonctionnalités :

- Créer une épargne libre
- Ajouter de l'argent (dépôt uniquement)
- Voir total épargné

B. Objectifs d'épargne

Fonctionnalités :

- Créer un objectif
- Définir montant cible et montant déjà épargné
- Calcul montant restant
- Barre de progression

5.6 Abonnements

Fonctionnalités :

- Ajouter / Modifier / Supprimer un abonnement
- Alertes si abonnement trop élevé
- Suggestions d'opportunité ("si tu investis ce montant...")
- Détection dépenses inutiles

5.7 Rapports

Fonctionnalités :

- Revenu total
- Dépense totale
- Solde de la période

Périodes :

- semaine
- mois
- année

Visualisation :

- Graphiques par catégorie
- Graphiques par compte

5.8 ChatBot (Assistant)

Fonctionnalités :

- Chat intégré
- Questions possibles :
 - dépenses sur une période
 - enveloppe restante
 - total épargné
 - plus grosses dépenses
 - résumé mensuel
- Conseils simples (budget, épargne, abonnements)

5.9 Paramètres

Fonctionnalités :

- Gestion profil :
 - photo
 - prénom
 - date de naissance
 - profession
 - pays
 - email
- Sécurité :
 - changement mot de passe avec authentification
- Notifications intelligentes :
 - création transaction
 - création compte
 - création enveloppe
 - création objectif épargne
 - création abonnement
 - enveloppe pleine
- Avancé :
 - mode sombre
 - changer devise
 - supprimer compte utilisateur
 - supprimer toutes les données

6. Navigation principale

La navigation principale de l'application contient les sections suivantes :

- a. Dashboard
- b. Transactions

- c. Comptes
- d. Enveloppes (Budget)
- e. Épargne
- f. Abonnements
- g. Rapports
- h. ChatBot
- i. Paramètres

7. Base de données (Firebase)

7.1 Firebase Authentication

Firebase Authentication est utilisé pour :

- Inscription des utilisateurs
- Connexion sécurisée
- Gestion des sessions
- Changement mot de passe

7.2 Cloud Firestore

Les données sont stockées dans Firestore sous forme de collections.

Exemple de structure :

- users
- accounts
- transactions
- envelopes
- savings
- savingGoals
- subscriptions
- notifications

Chaque document est lié à un utilisateur via son identifiant Firebase (uid).

8. Phase actuelle : Tests & validation

8.1 Objectif de la phase de test

La phase de test vise à :

- vérifier le bon fonctionnement de toutes les fonctionnalités
- détecter et corriger les bugs
- améliorer l'expérience utilisateur
- garantir la stabilité de l'application avant livraison

9. Calendrier de réalisation (planning)

Période	Activités
Séance 1	Analyse + cahier des charges
Séance 2	Maquettes + conception base de données
Séance 3	création de la charte graphique (Vert néon sur Noir)
Séance 4	Codage de la page d'accueil (Hero Section) avec l'animation de l'image qui flotte.
Séance 5	Intégration du menu de navigation fixe avec effet de flou (Glassmorphism).
Séance 6	Création de la section "Features" avec les cartes interactives (Gestion, Épargne, Sécurité).
Séance 7	Configuration de la console Firebase et liaison avec le projet JavaScript.
Séance 8	Création du formulaire d'inscription et gestion du stockage des utilisateurs dans Firestore.
Séance 9	Développement de la page de connexion (Login) et gestion de la session avec localStorage.
Séance 10	Mise en place de la protection des pages (redirection si l'utilisateur n'est pas connecté).
Séance 11	Design du tableau de bord principal et création du bouton flottant (FAB) d'ajout.
Séance 12	Développement du formulaire modal pour ajouter une transaction (Dépense/Revenu).
Séance 13	Codage de la fonction CRUD : affichage des transactions dans un tableau dynamique.
Séance 14	Logique des soldes : mise à jour automatique des comptes (TMoney, Cash, Banque) selon les transactions.
Séance 15	Système de catégories et liaison avec les enveloppes budgétaires pour limiter les dépenses.
Séance 16	Développement des fonctions de modification et de suppression des transactions (avec impact sur le solde).
Séance 17	Mise en place du moteur de recherche et des filtres par date/compte (Correction des bugs d'alignement).
Séance 18	Mise en place de la section comptes avec les fonctionnalité d'ajout et de modification
Séance 19	Mise en place de la section enveloppes budgétaire avec les fonctionnalité d'ajout et de modification et la

	logique
Séance 20	Mise en place de la section objectifs eparnes avec les fonctionnalité d'ajout et de modification et du chatbot et enfin la section d'abonnement

10. Livrables attendus et situation actuelle

10.1 Liste des livrables

Livrable	Description
Cahier des charges	Document des besoins
Code source	Projet complet
Base de données Firebase	Collections + règles
Maquettes / UI	Interface finale
Rapport de réunions	PV des rencontres tuteur
Planning / calendrier	Organisation des tâches
Guide de déploiement	Installation et lancement
Guide d'utilisation	Manuel utilisateur
Rapport final	Document final
Tests & corrections	Validation + corrections

11. Bases de données

- **Collection : users**

Rôle : Stocke les informations personnelles de chaque utilisateur inscrit.

Fichiers concernés : auth.js, dashboard-auth.js

ID du document : L'UID fourni par Firebase Authentication.

Champs :

- nom (string) : Nom de famille.
- prenom (string) : Prénom.
- email (string) : Email de l'utilisateur.
- pays (string) : Pays de résidence.
- telephone (string) : Numéro de téléphone.
- profession (string) : Profession.
- currency (string) : Devise par défaut (ex: "FCFA").
- createdAt (Timestamp) : Date de création du compte.
- photoURL (string, optionnel) : URL de l'avatar (pour les connexions sociales).
- role (string, optionnel) : Rôle de l'utilisateur (ex: "admin").

- **Collection : les comptes**

Rôle : Gère les différents comptes financiers de l'utilisateur (bancaire, mobile money, cash).

Fichier concerné : accounts.js

ID du document : Un ID unique auto-généré par Firestore.

Champs :

- uid (string) : Lien vers le document de l'utilisateur dans la collection users.
- name (string) : Nom du compte (ex: "Banque Atlantique", "T-Money", "Espèces").
- balance (number) : Solde actuel du compte.
- type (string) : Type de compte (ex: 'bank', 'tmoney', 'cash').
- number (string, optionnel) : Numéro de compte ou de téléphone associé.

- **Collection : transactions**

Rôle : Enregistre chaque entrée (revenu) et sortie (dépense) d'argent.

Fichiers concernés : abonnements.js (création), accounts.js (lecture). Le fichier principal transactions.js n'est pas fourni mais il est le consommateur principal de cette collection.

ID du document : Un ID unique auto-généré par Firestore.

Champs :

- uid (string) : Lien vers le document de l'utilisateur.
- account (string) : Nom du compte où la transaction a eu lieu.
- amount (number) : Montant de la transaction.
- type (string) : 'income' (revenu) ou 'expense' (dépense).
- desc (string) : Description de la transaction (ex: "Achat supermarché").
- category (string) : Catégorie de la dépense (ex: "Alimentation", "Transport").
- date (string) : Date de la transaction (format "YYYY-MM-DD").
- createdAt (Timestamp) : Date d'enregistrement dans la base.

- **Collection : les abonnements**

Rôle : Suit les abonnements mensuels récurrents de l'utilisateur.

Fichier concerné : abonnements.js

ID du document : Un ID unique auto-généré par Firestore.

Champs :

- uid (string) : Lien vers le document de l'utilisateur.
- name (string) : Nom de l'abonnement (ex: "Netflix", "Canal+").
- amount (number) : Coût mensuel de l'abonnement.
- day (number) : Jour du mois où le prélèvement a lieu.
- lastPaymentMonth (string) : Marqueur pour savoir si le paiement du mois en cours a été simulé (ex: "2024-4").

- **Collection : notifications**

Rôle : Stocke les notifications internes à l'application pour un utilisateur.

Fichier concerné : dashboard-auth.js

ID du document : Un ID unique auto-généré par Firestore.

Champs :

- uid (string) : Lien vers le document de l'utilisateur.
- title (string) : Titre de la notification.
- message (string) : Contenu du message.
- type (string) : Type de notification ('success', 'info', 'error').
- read (boolean) : true si l'utilisateur a lu la notification, sinon false.
- date (string) : Date de création au format ISO.

- **Collection : les objectifs d'épargne**

Rôle : Permet de définir et suivre des cagnottes pour des projets futurs (ex: "Voyage", "Voiture") ou une épargne de précaution.

Fichier concerné : budgets.js

ID du document : Un ID unique auto-généré par Firestore (ou general_{uid} pour l'épargne libre).

Champs :

- uid (string) : Lien vers le document de l'utilisateur.
- name (string) : Nom de l'objectif (ex: "Vacances").
- target (number) : Montant cible à atteindre (0 pour une épargne libre).
- current (number) : Montant actuellement épargné.
- dueDate (string) : Date d'échéance souhaitée (format "YYYY-MM-DD").
- color (string, optionnel) : Code couleur pour l'affichage (utilisé notamment pour l'épargne libre).

- **Collection : les enveloppes budgétaires**

Rôle : Permet de plafonner les dépenses mensuelles par catégorie pour contrôler le budget et éviter les dépassements.

Fichier concerné : budgets.js

ID du document : Un ID unique auto-généré par Firestore.

Champs :

- uid (string) : Lien vers le document de l'utilisateur.
- category (string) : Catégorie de dépense concernée (ex: "Alimentation", "Loisirs" ou catégorie personnalisée).
- limit (number) : Plafond de dépense autorisé pour le mois.
- spent (number) : Cumul des dépenses déjà effectuées dans cette catégorie.

12. Guide de déploiement

Étape 1 : Préparation des fichiers

Les serveurs web cherchent par défaut un fichier nommé index.html pour la page d'accueil. Dans notre projet, notre page d'accueil est refresh.html.

Donc dans notre dossier frondend.

On Renomme le fichier refresh.html en index.html.

Étape 2 : Installation des outils Firebase

on aura besoin de Node.js installé sur notre ordinateur.

Ensuite, dans le terminal (Invite de commande ou PowerShell) on installe l'outil CLI de Firebase :

```
bash
npm install -g firebase-tools
```

Étape 3 : Initialisation du projet

Dans notre terminal, on se place à la racine de notre projet (le dossier parent de frondend) :

```
bash
cd "c:\Users\DG\Documents\Cours Semestre5\Miabe_Budet_Manager"
```

on se connecte à notre compte Google :

```
bash
firebase login
```

Initialisez le projet :

```
bash
firebase init
```

Répondez aux questions suivantes :

Which Firebase features do you want to set up? on selectionne Hosting: Configure files for Firebase Hosting and (optionally) set up GitHub Action deploys

Please select an option: on selectionne Use an existing project et sélectionnez votre projet Firebase actuel dans la liste.

What do you want to use as your public directory? On tape : frondend .

Configure as a single-page app (rewrite all urls to /index.html)? on selectionne : No

Set up automatic builds and deploys with GitHub? on selectionne: No

File frondend/index.html already exists. Overwrite? on selectionne: No

Étape 4 : Déploiement

Une fois l'initialisation terminée, on lance simplement :

```
bash
```

```
firebase deploy
```

La console va travailler un peu et affichera à la fin : Hosting URL: <https://votre-projet-id.web.app>

C'est l'adresse du site en ligne !

Étape 5 : Configuration Post-Déploiement

Maintenant que notre site a une nouvelle adresse (ex: <https://miabe-manager.web.app>), nous devons dire à Firebase que cette adresse est sécurisée pour l'authentification.

sur la Console Firebase (sur le web).

Menu Authentication -> Onglet Settings (Paramètres).

dans Authorized domains (Domaines autorisés).

On ajoute le nouveau domaine (ex: miabe-manager.web.app).

Étape 6 : Sécuriser vos données (Règles Firestore)

Comme notre code est maintenant public, il faut s'assurer que personne ne peut voler les données des autres.

dans la Console Firebase -> Firestore Database -> Onglet Règles et on colle ceci :

```
javascript
```

```
Show full code block
```

```
rules_version = '2';
```

```
service cloud.firestore {
```

```
  match /databases/{database}/documents {
```

```
    // Fonction helper pour vérifier si l'utilisateur est connecté et accède à ses propres données
```

```
    function isOwner(userId) {
```

```
      return request.auth != null && request.auth.uid == userId;
```

```
    }
```

```
    // Règles pour les utilisateurs
```

```
    match /users/{userId} {
```

```
      allow read: if isOwner(userId) || (request.auth != null &&
```

```

get(/databases/$(database)/documents/users/$(request.auth.uid)).data.role ==
'admin');
  allow write: if isOwner(userId); // Chacun modifie son propre profil
}

// Règles pour les collections liées à l'UID (comptes, transactions, budgets, etc.)
match /{collection}/{document=**} {
  allow read, write: if request.auth != null && resource.data.uid ==
request.auth.uid;
  allow create: if request.auth != null && request.resource.data.uid ==
request.auth.uid;
}
}
}

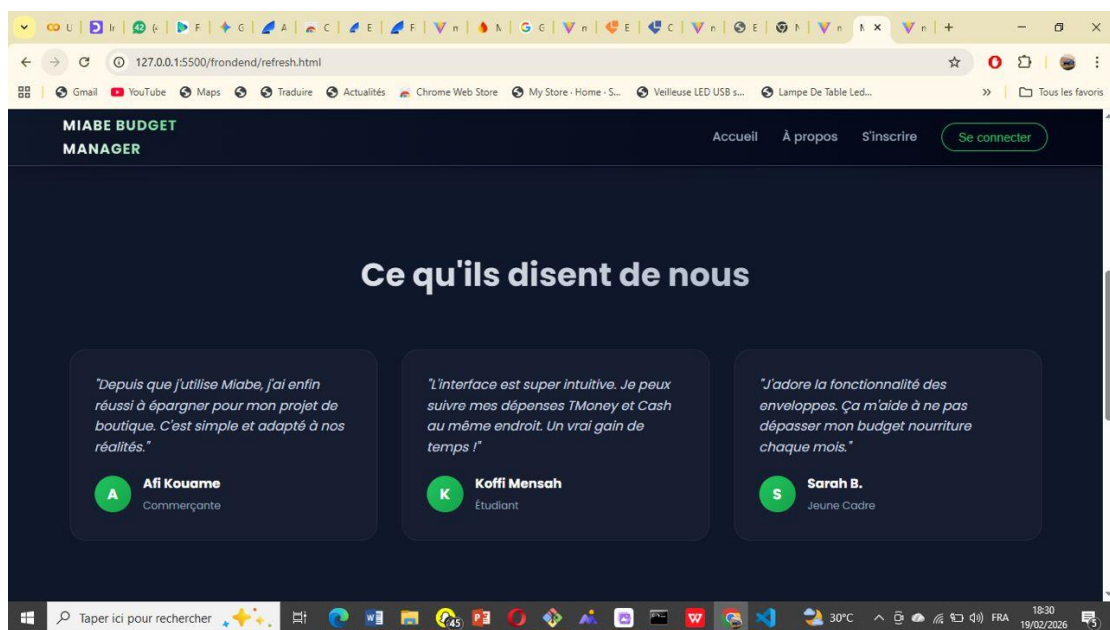
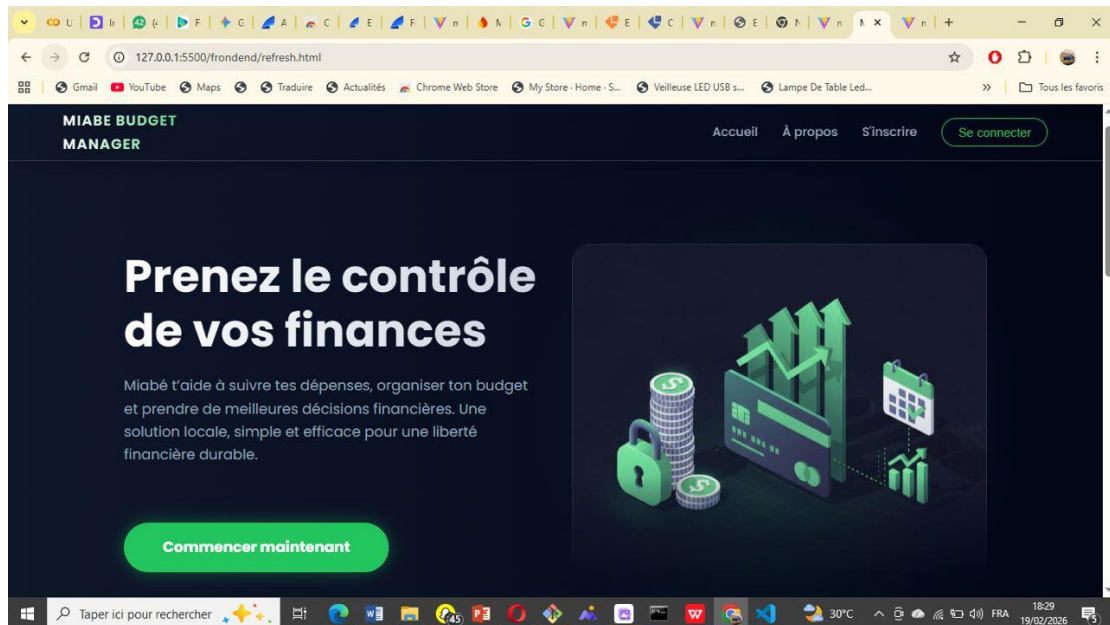
```

13. Guide d'utilisation

La Page d'Accueil & Communauté

La porte d'entrée de ton indépendance financière :

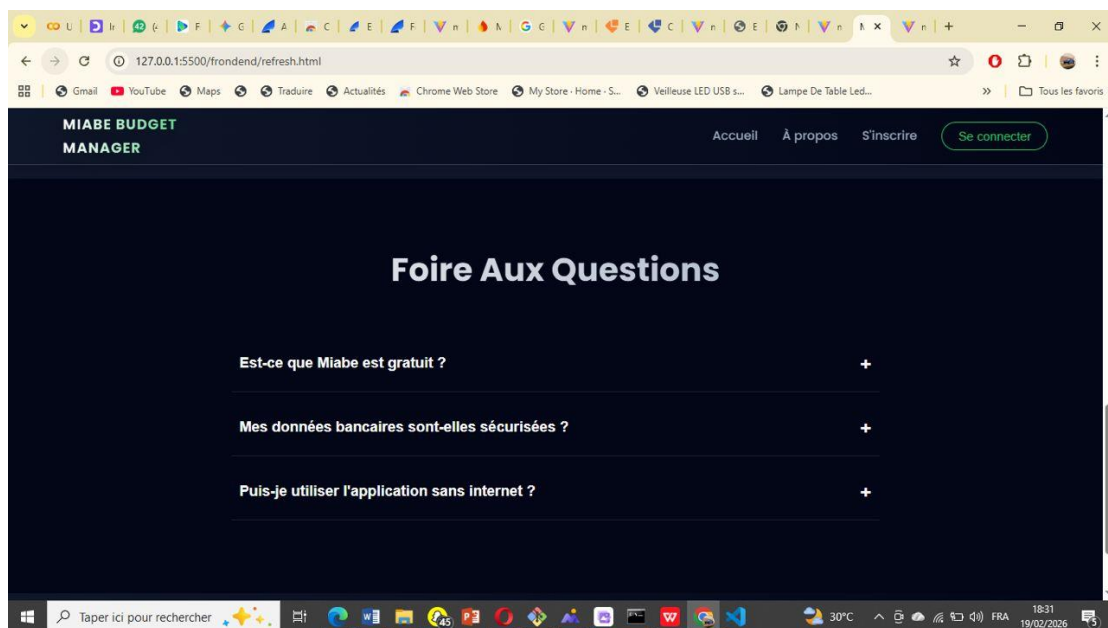
- **Présentation:** Un aperçu rapide de ta situation avec le bouton "Commencer maintenant".
- **Ce qu'ils disent de nous :** Retrouvez les témoignages d'utilisateurs comme Afi, Koffi ou Sarah qui partagent comment Miabé a changé leur gestion quotidienne.
- **FAQ :** Une foire aux questions interactive pour répondre à vos doutes sur la sécurité et les coûts.



Foire Aux Questions (FAQ)

Si vous avez des doutes, consultez la section FAQ sur la page d'accueil :

- **Gratuité** : Miabé est-il gratuit ? Oui !
- **Sécurité** : Vos données bancaires sont chiffrées et sécurisées.
- **Mode Hors-ligne** : Des informations sur l'utilisation sans internet sont également disponibles.

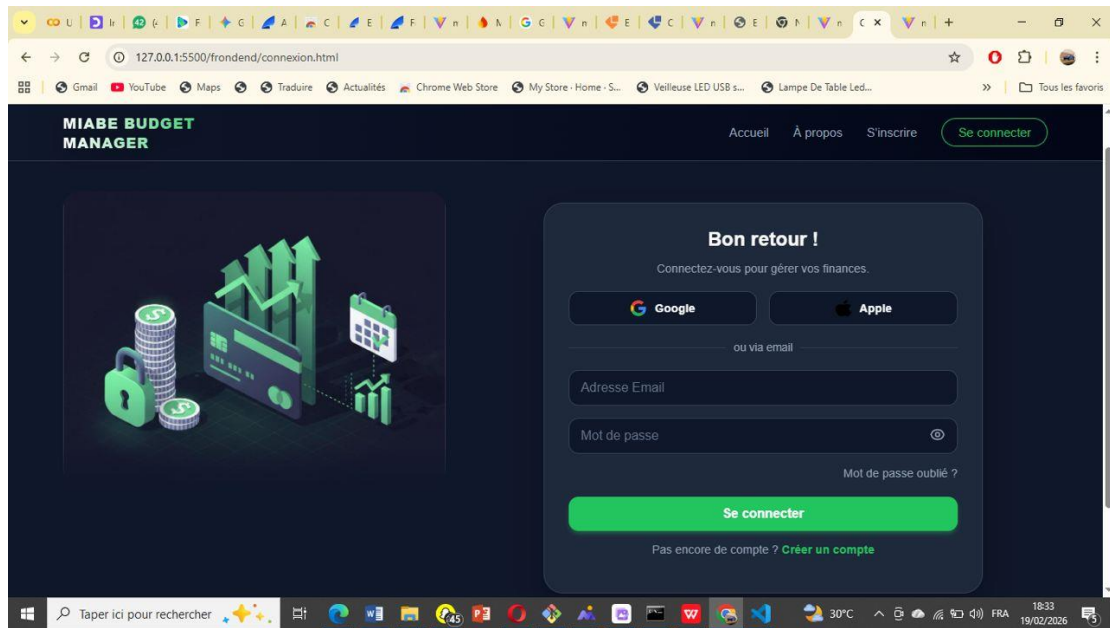


Connexion :

Accès et Inscription (Validation Admin)

Pour garantir la sécurité de la communauté, le processus d'inscription est contrôlé :

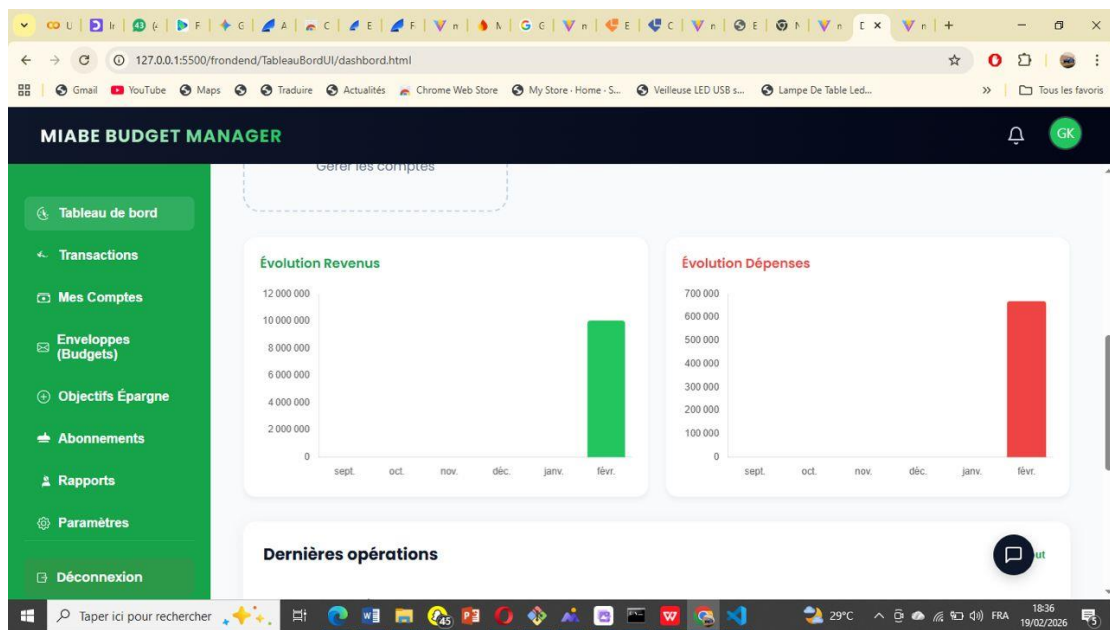
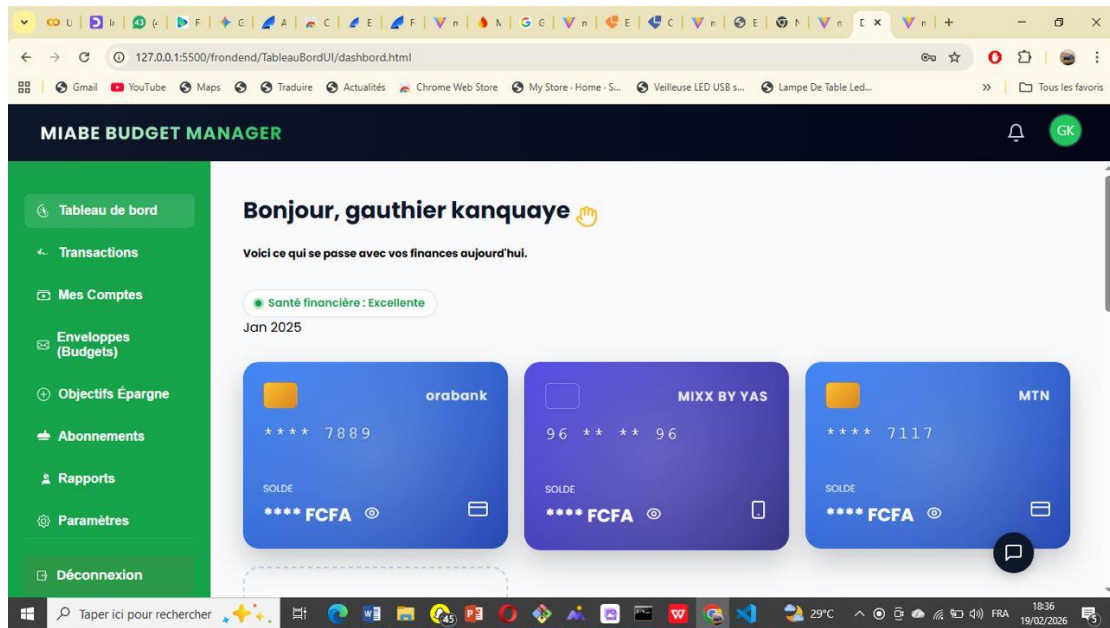
1. **Création de compte** : Allez sur la page d'inscription et remplissez vos informations.
2. **Validation par l'Administrateur** : Une fois votre formulaire envoyé, votre compte doit être validé par l'administrateur. Vous ne pourrez pas vous connecter tant que cette étape n'est pas franchie.
3. **Connexion** : Une fois validé, utilisez vos identifiants sur la page "Bon retour !" pour accéder à votre tableau de bord.

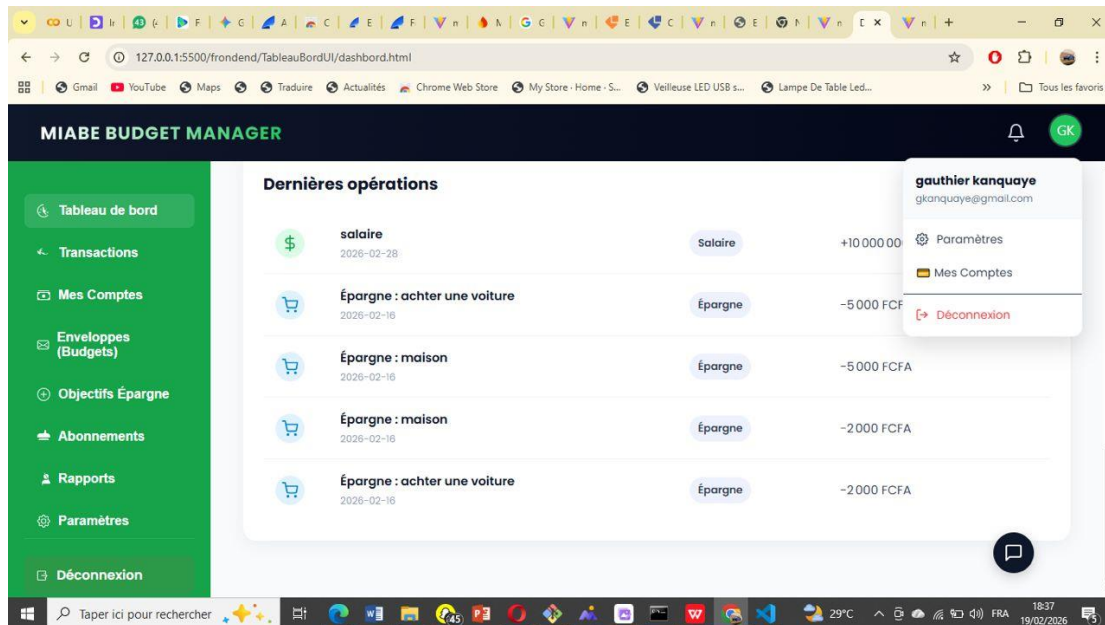


Le Tableau de Bord (Vue Globale)

Votre centre de commande pour une vision claire de votre argent :

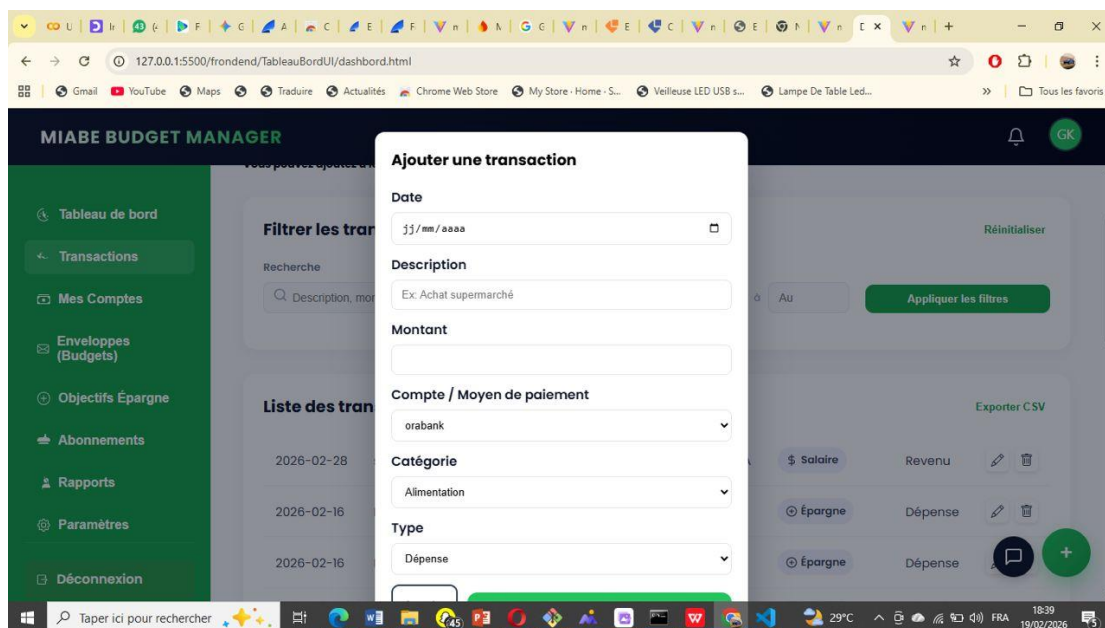
- **Vue d'ensemble** : Un résumé total de tous vos comptes combinés.
- **Graphiques** : Analyse visuelle de vos revenus par rapport à vos dépenses pour suivre l'évolution de votre santé financière.
- **Dernières activités** : Affichage rapide des **10 dernières transactions** pour un suivi immédiat.
- **Notifications & Profil** : Une cloche pour vos alertes et un accès rapide à votre profil via vos initiales.

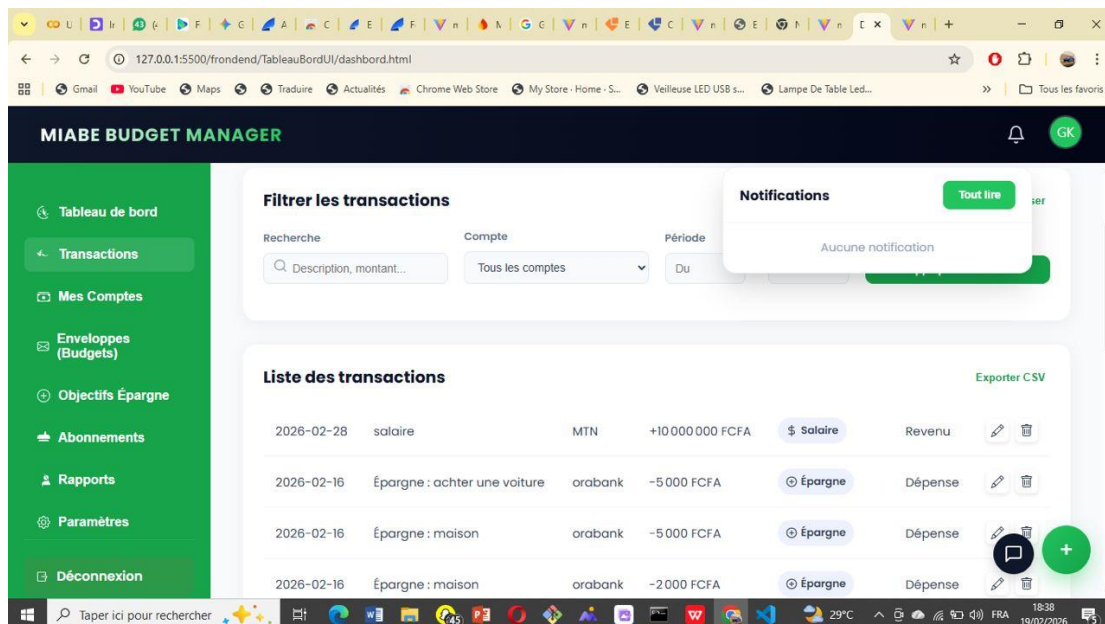




Transactions :

- **Transactions** : Le module principal pour enregistrer vos entrées et sorties. Vous pouvez filtrer par date ou par compte pour retrouver un paiement précis.
- **Ajouter une transaction** : Cliquez sur "+" pour ajouter vos transaction et définir votre solde actuel.

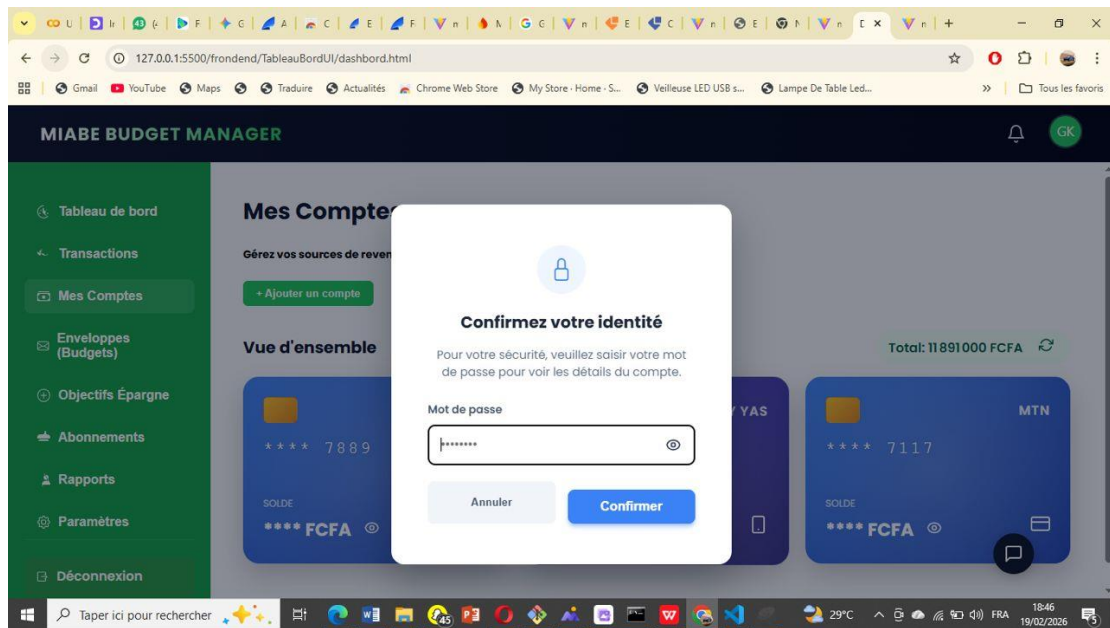
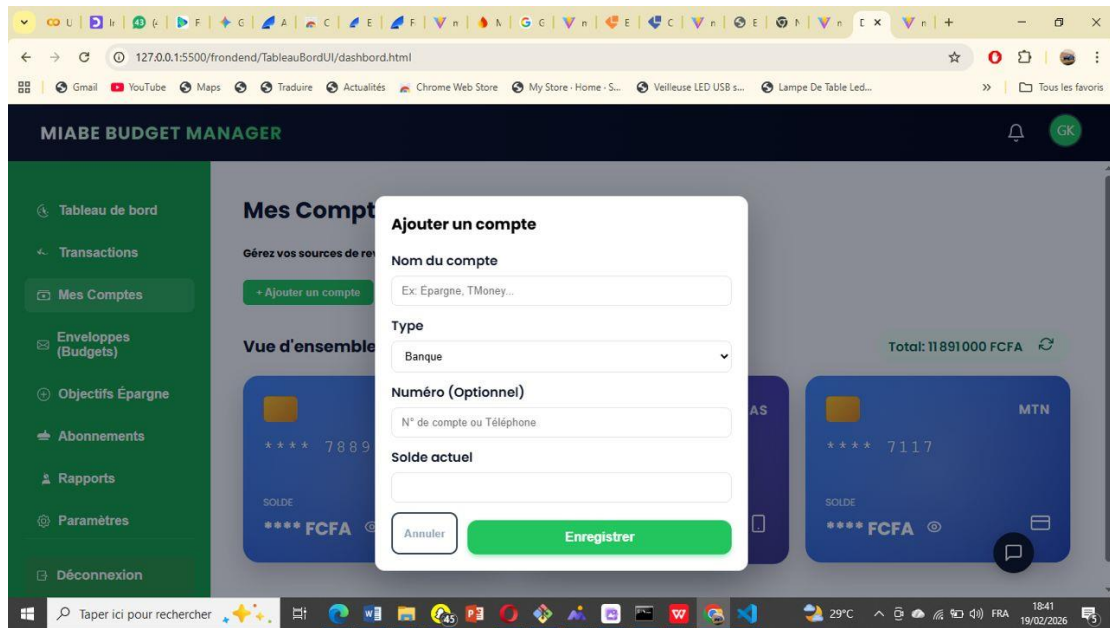


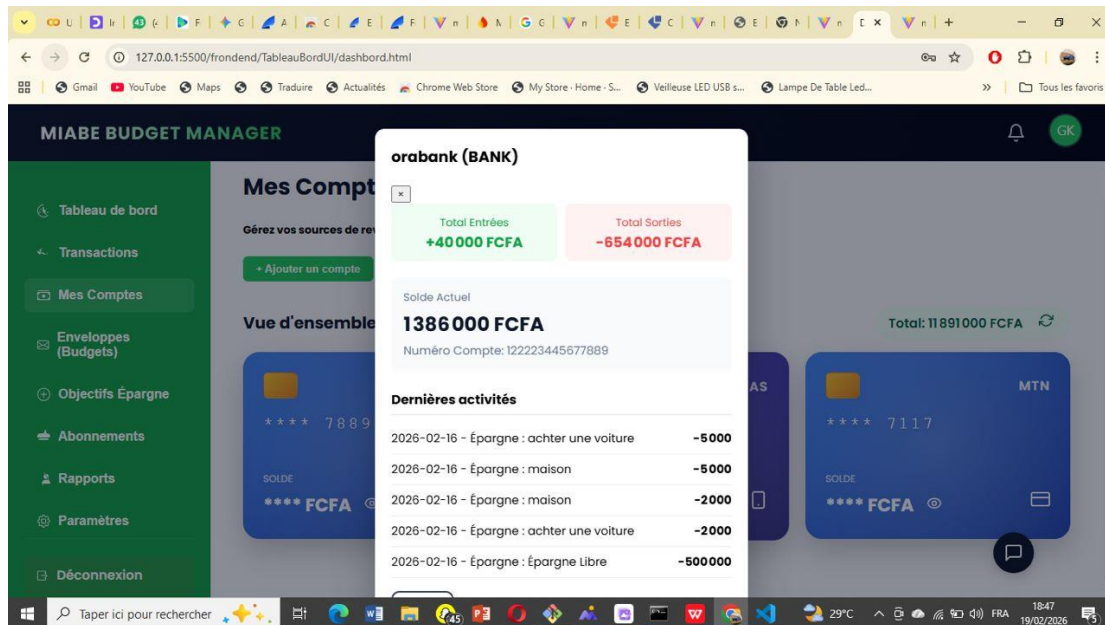


Gestion de vos Comptes

Le cœur de l'application vous permet de centraliser toutes vos sources d'argent :

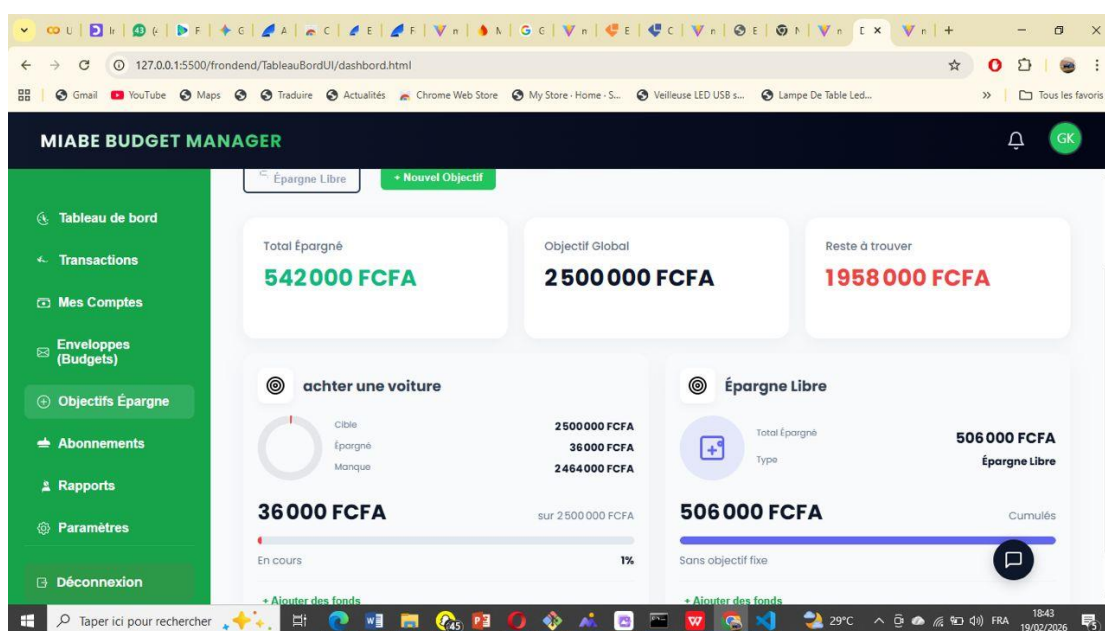
- **Ajouter un compte** : Cliquez sur "+ Ajouter un compte" pour configurer vos comptes (Banque, TMoney, etc.) et définir votre solde actuel.
- **Sécurité renforcée** : Pour consulter les détails sensibles d'un compte, l'application vous demandera de **confirmer votre identité** en saisissant à nouveau votre mot de passe.
- **Analyse du compte** : Dans la vue détaillée, vous pouvez voir le total des entrées, des sorties et l'historique de vos dernières activités.

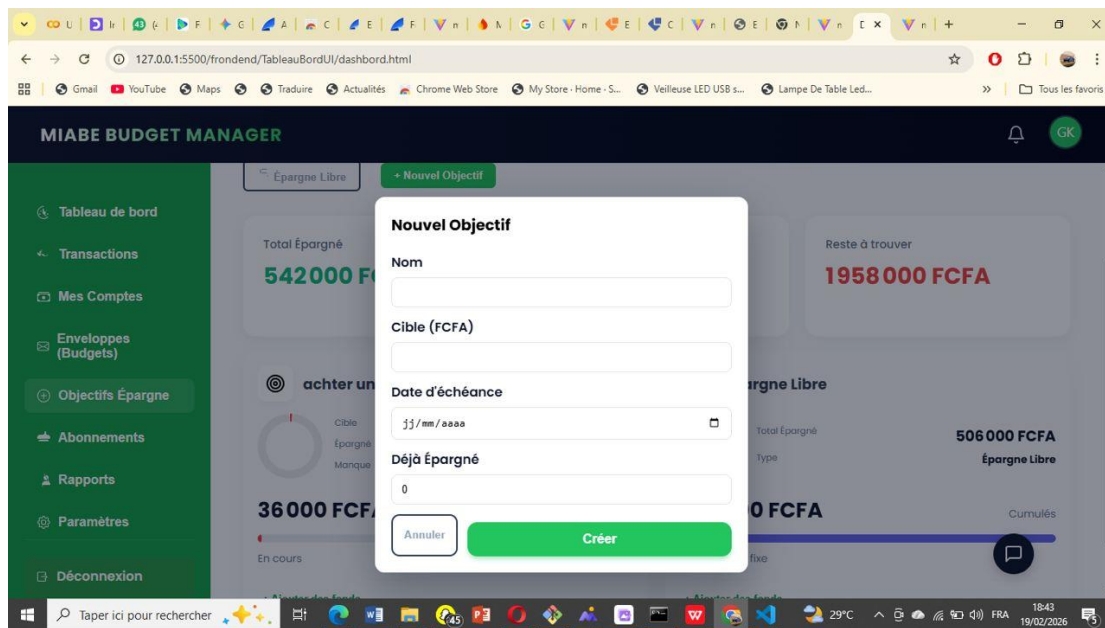




Objectifs d'Épargne

- **Objectifs d'Épargne :** Définissez des buts concrets (achat de voiture, projet immobilier) et suivez votre progression.
- **Ajouter un objectif :** Cliquez sur "+ Ajouter objectif" pour configurer vos objectifs (voiture, iphone, etc.) et définir votre solde actuel.
- **Détails d'Épargne :** Dans la vue compte, vous pouvez voir exactement quelle somme a été allouée à chaque projet (ex: -2000 FCFA pour "maison").

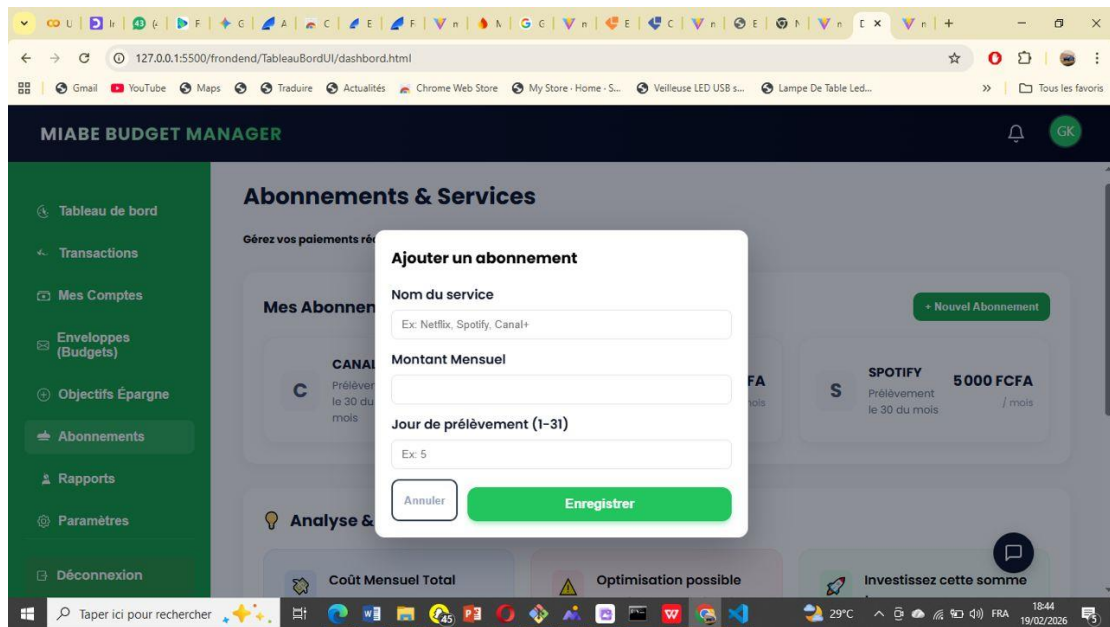
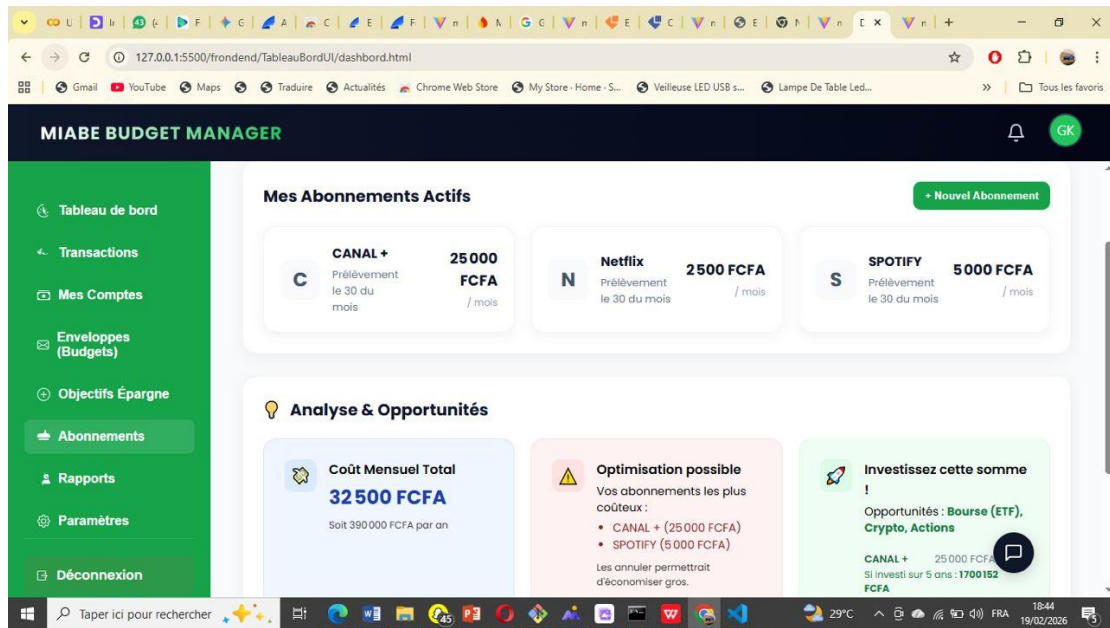




Abonnements et Services

Gérez vos paiements récurrents pour éviter les surprises :

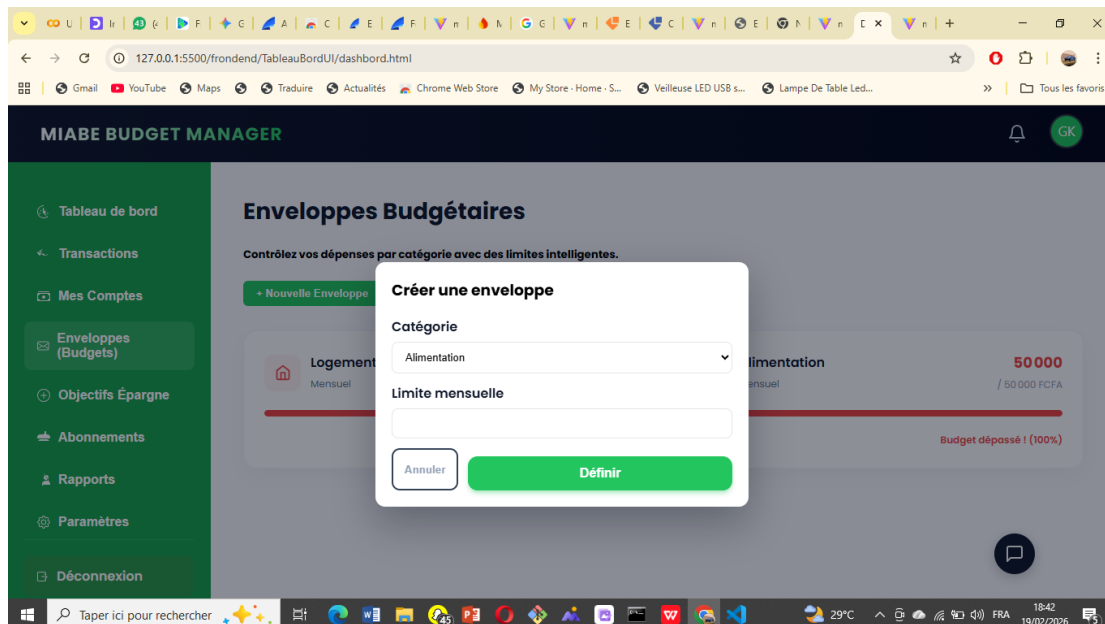
- **Suivi des services** : Enregistrez vos abonnements comme Canal+, Netflix ou Spotify.
- **Ajouter un abonnement** : Cliquez sur "+ Nouvel abonnement " pour configurer vos abonnements (netflix, canal+, etc.).
-
- **Analyse du coût** : L'application calcule votre **Coût Mensuel Total** et le projette sur l'année.
- **Optimisation** : Consultez la section "Analyse & Opportunités" pour identifier vos abonnements les plus coûteux et recevoir des conseils d'investissement.



Enveloppes et Budgets

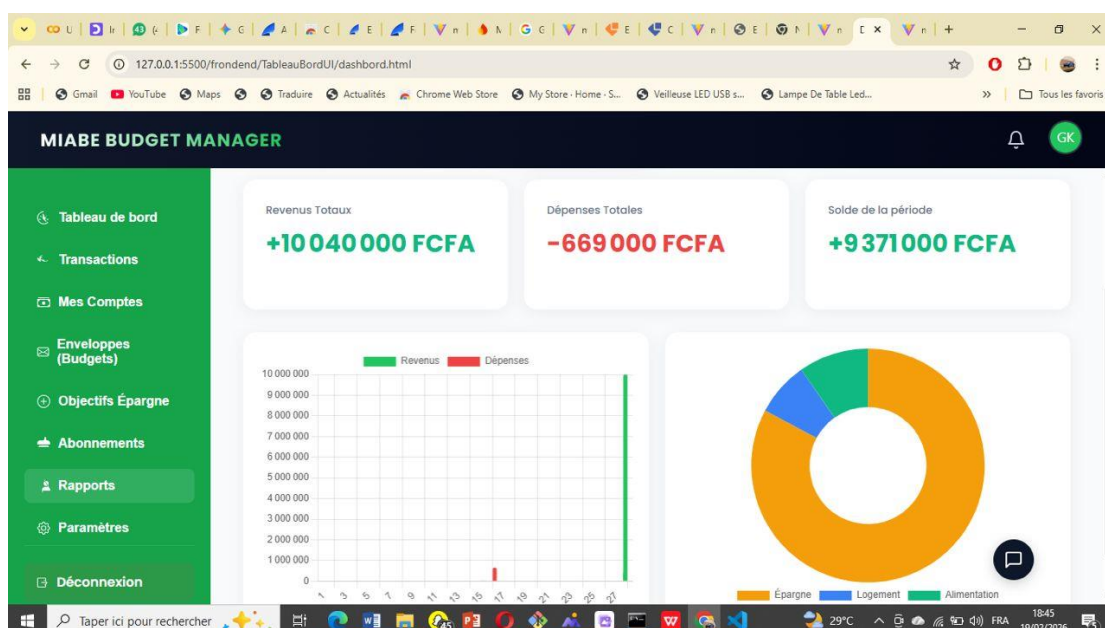
Ne dépassez plus jamais vos limites grâce aux enveloppes intelligentes :

- **Créer une enveloppe** : Choisissez une catégorie (ex: Alimentation) et fixez une limite mensuelle.
- **Suivi en temps réel** : Une barre de progression vous indique où vous en êtes. Si vous dépassez la limite, l'application affiche une alerte "Budget dépassé !" en rouge.



Rapports :

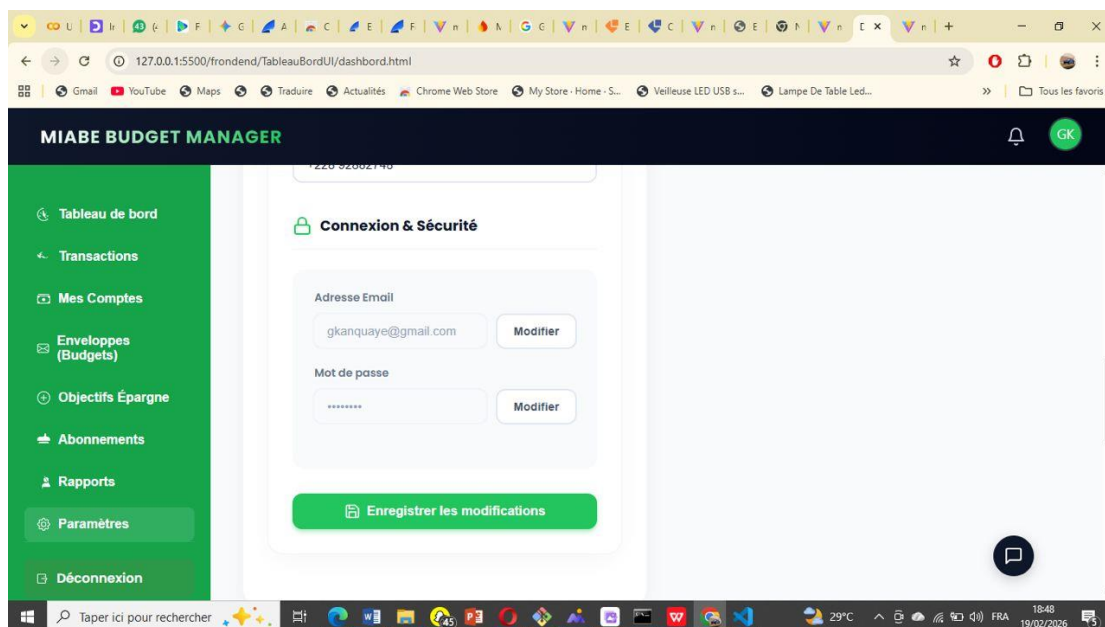
Une vue d'ensemble de votre santé financière.

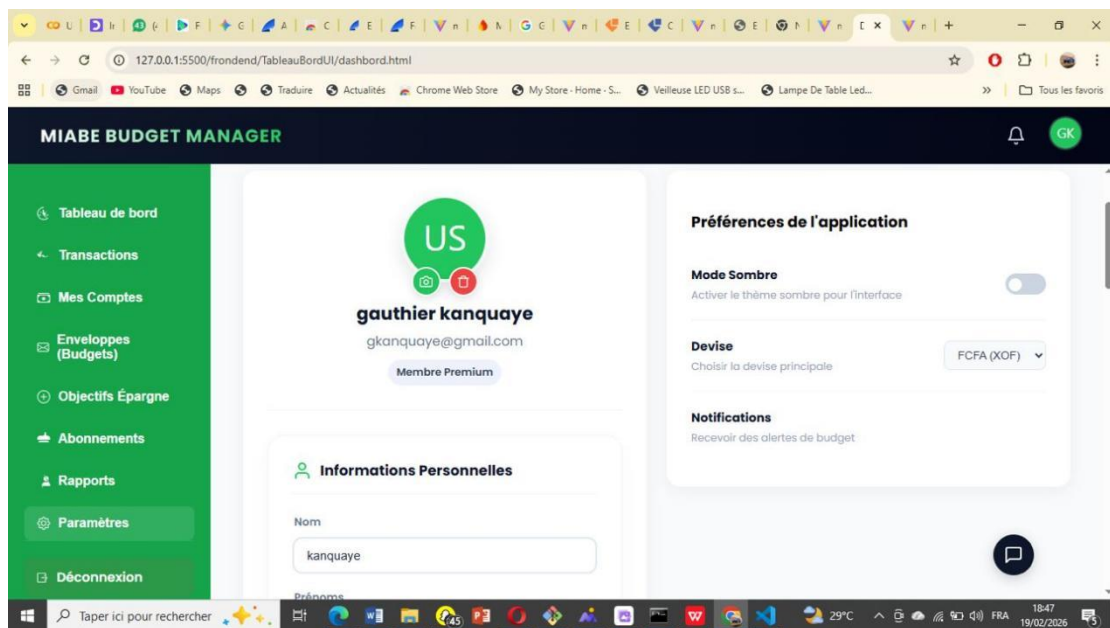
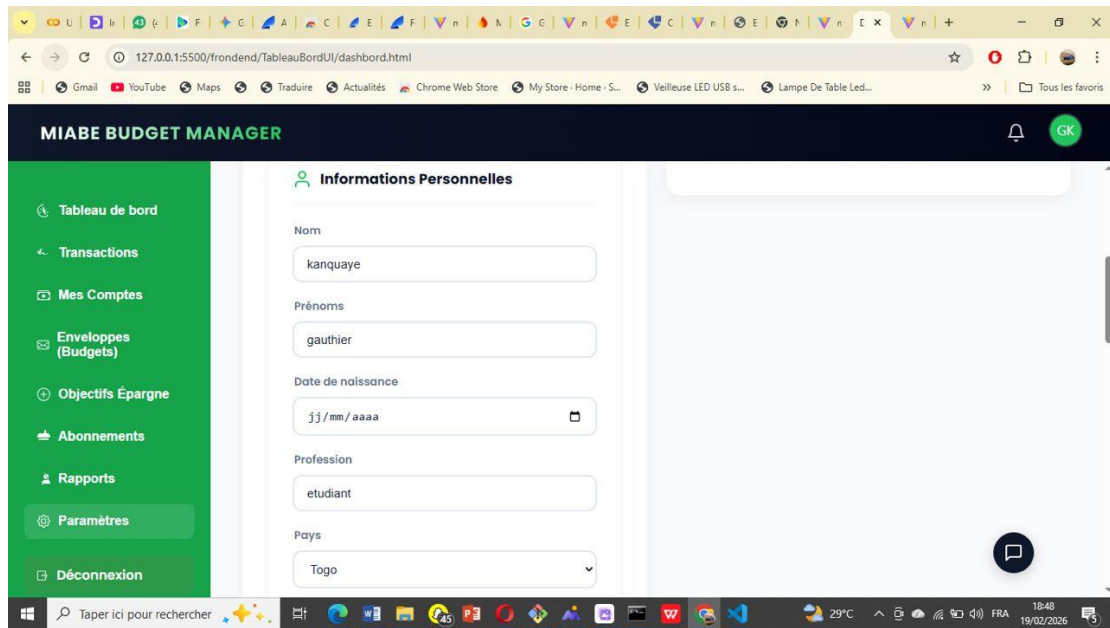


Paramètres et Personnalisation

L'espace pour configurer votre expérience en toute sécurité :

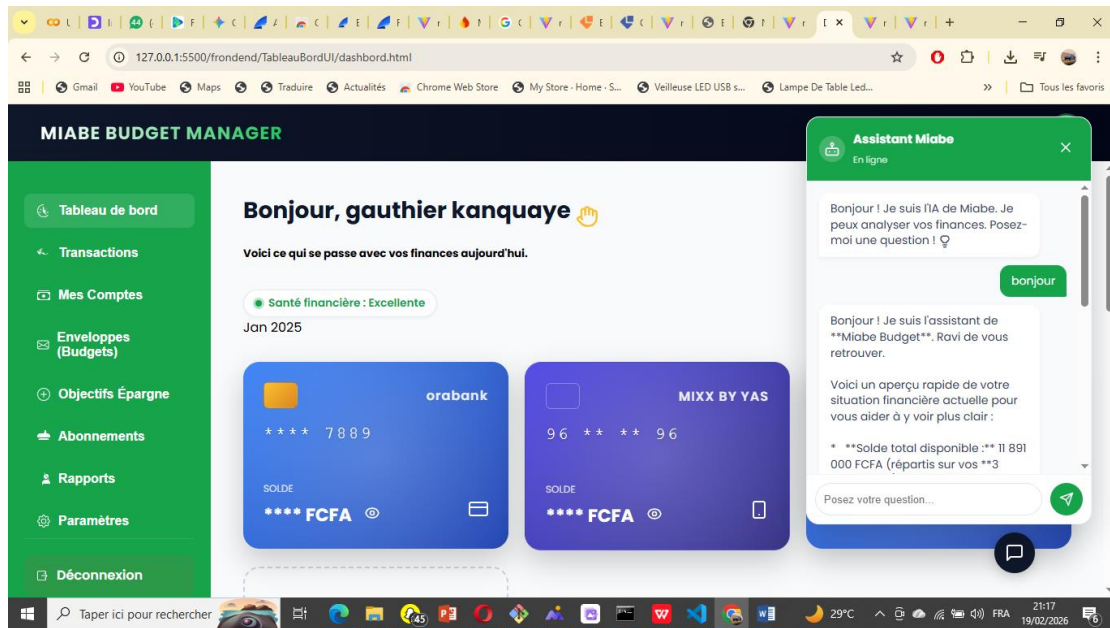
- **Mode Sombre** : Activez ou désactivez le Dark Mode selon vos préférences visuelles.
- **Sécurité du compte** : Changez votre **mot de passe** ou votre **email** (une authentification préalable est requise pour ces modifications).
- **Gestion des données** : Possibilité de supprimer vos informations personnelles ou de clôturer définitivement votre compte.





Assistant Intelligent (Chatbot) :

Une bulle de discussion en bas à droite vous permet de poser vos questions à tout moment pour obtenir de l'aide.



14. Difficultés rencontrées

- Gestion des règles Firestore (sécurité par utilisateur)
- Synchronisation des données en temps réel
- Gestion des filtres et exports
- Optimisation de l'interface (responsive)
- Mise en place des graphiques et statistiques

15. Conclusion

Le projet Miabe Budget Manager est désormais à un stade avancé et toutes les fonctionnalités prévues ont été intégralement développées avec succès. L'équipe se concentre actuellement sur une phase de tests rigoureux visant à garantir la fiabilité et la performance optimale de l'application. Parallèlement, les derniers ajustements sont effectués pour corriger les éventuels bugs restants, tandis que les guides de déploiement et d'utilisation sont finalisés afin d'assurer une prise en main fluide et efficace. Ces étapes clés permettront de préparer une livraison finale de haute qualité, répondant pleinement aux attentes.