Application de suivi fitness

Rockson Noel
FITNESS_S.A | **218, AV. JEAN PAUL 2, HAUT TURGEAU, PORT-AU-PRINCE, HAÏTI**

1 Résumé exécutif

Ce projet vise à développer une application de suivi de fitness qui permettra aux utilisateurs de suivre leurs activités physiques, définir des objectifs personnels et consulter des statistiques sur leurs progrès. L'application sera conviviale, attrayante et offrira une expérience utilisateur fluide.

1.1 Objectifs du projet

- Suivi des activités : Permettre aux utilisateurs de suivre leurs séances d'entraînement, incluant le type d'exercice, la durée, et l'intensité.
- Personnalisation des objectifs : Les utilisateurs peuvent définir des objectifs spécifiques (ex : perdre du poids, améliorer l'endurance) et suivre leur progression.
- Analyse des performances : Fournir des rapports et des visualisations des données pour aider les utilisateurs à comprendre leurs performances et à ajuster leur plan d'entraînement.
- Fonctionnalités sociales : Offrir des options pour partager les progrès avec des amis, participer à des défis, et recevoir des encouragements.

1.2 Public cible

- **Sportifs Amateurs et Professionnels** : Ceux qui souhaitent suivre et améliorer leurs performances.
- **Personnes en Quête de Bien-Être** : Utilisateurs souhaitant améliorer leur condition physique et suivre leurs activités quotidiennes.

1.3 Contexte et justification

Avec l'essor des technologies mobiles et l'importance croissante du bien-être, il existe un besoin accru d'outils permettant de suivre les activités physiques et les progrès. Cette application répondra à ce besoin en fournissant une solution complète pour la gestion du fitness, avec des fonctionnalités avancées et une interface intuitive.

2 Portée du Projet

2.1 Fonctionnalité principales

1. **Authenfication et Gestion des Utilisateurs**:

- Inscription et connexion via email et mot de passe.
- Authentification via des services tiers (Google, Facebook).

2. **Suivi des Séances d'Entraînement**:

    - Enregistrement manuel des séances (type d'exercice, durée, calories brûlées).

- Suivi automatique via des capteurs de l'appareil (GPS pour la course, accéléromètre pour les pas).
- Historique des entraînements avec des filtres par type, date, etc.

3. **Définition et Suivi des Objectifs de Fitness**:

    - Définition d'objectifs personnels (par exemple, courir 5 km, perdre 5 kg, etc.).
    - Suivi des progrès vers ces objectifs avec des graphiques et des notifications de rappel.
    - Récompenses et badges pour atteindre des objectifs.

4. **Visualisation des Statistiques et des Progrès**:

    - Tableaux de bord avec des graphiques (courbes, histogrammes) pour visualiser les progrès sur différentes périodes (jour, semaine, mois).
    - Analyse des tendances et des performances (meilleurs temps, augmentation des poids soulevés, etc.).
    - Comparaison des performances avec des périodes précédentes.

5. **Notifications et Rappels**:

    - Notifications push pour rappeler les objectifs quotidiens ou hebdomadaires.
    - Rappels pour les séances d'entraînement planifiées.
    - Notifications pour les nouveaux badges ou récompenses obtenus.

6. **Interface Utilisateur Attrayante et Intuitive**:

    - Design moderne et responsive.
    - Navigation intuitive avec une barre de navigation ou un menu latéral.
    - Utilisation de graphiques et d'animations pour une meilleure visualisation des données.

7. **Fonctionnalités Sociales (Optionnel)**:

    - Partage des progrès sur les réseaux sociaux.
    - Défis et compétitions avec des amis.
    - Commentaires et encouragements entre utilisateurs.

2.2 Limitations

1. **Support des Capteurs**:

    - L'application dépend des capteurs intégrés dans les appareils des utilisateurs pour le suivi automatique des activités. Cela peut limiter la précision et la disponibilité de certaines fonctionnalités sur des appareils plus anciens ou moins avancés.

2. **Dépendance aux Services Tiers**:

- L'application utilise Firebase pour l'authentification, les notifications et le stockage des fichiers. Toute interruption ou modification de ces services peut affecter le fonctionnement de l'application.

3. **Confidentialité et Sécurité**:

    - La gestion des données personnelles et de santé des utilisateurs nécessite une attention particulière à la sécurité et à la confidentialité. Cela inclut le respect des réglementations locales et internationales sur la protection des données.

4. **Complexité de l'Intégration**:

    - L'intégration des fonctionnalités sociales et des défis entre amis peut nécessiter des développements supplémentaires et des accords avec les plateformes de réseaux sociaux

3 Planification

3.1 Chronologie des étapes de développement

La planification détaillée du projet est essentielle pour garantir une exécution en temps voulu et dans les limites du budget. Voici les phases principales et leur durée estimée :

1. **Phase 1 : Planification et Conception (1 mois)**

    - **Semaine 1** : Réunion de lancement, définition des objectifs, analyse des besoins.
    - **Semaine 2** : Conception de l'architecture technique et de l'interface utilisateur.
    - **Semaine 3** : Création des maquettes et prototypes.
    - **Semaine 4** : Validation des maquettes et ajustements.

2. **Phase 2 : Développement Frontend et Backend (3 mois)**

    - **Mois 1** : Développement de l'application mobile avec Flutter, création des écrans de base.
    - **Mois 2** : Développement de l'application web avec React.js/Angular, intégration de l'authentification.
    - **Mois 3** : Développement du backend avec Node.js et Express.js, création des API REST.

3. **Phase 3 : Intégration des Services et Base de Données (2 mois)**

    - **Mois 1** : Intégration de Firebase pour l'authentification et les notifications.
    - **Mois 2** : Configuration de Firestore et MongoDB pour le stockage des données.

4. **Phase 4 : Tests et Déploiement (1 mois)**

    - **Semaine 1** : Tests unitaires et d'intégration.

- **Semaine 2** : Tests utilisateurs et feedback.
- **Semaine 3** : Correction des bugs et optimisation.
- **Semaine 4** : Préparation et déploiement de la version finale.

3.2 Répartitions des taches

Pour garantir une exécution efficace et ordonnée, les tâches sont réparties entre différentes équipes spécialisées.

1. **Équipe A : Développement Frontend**
   - **Tâches** :
     - Développement de l'application mobile (Flutter)
     - Développement de l'application web (React.js/Angular)
     - Conception de l'interface utilisateur
   - **Responsables** : Développeurs frontend, designers UX/UI
2. **Équipe B : Développement Backend**
   - **Tâches** :
     - Développement du serveur backend (Node.js, Express.js)
     - Création des API REST
     - Gestion de la base de données (Firestore, MongoDB)
   - **Responsables** : Développeurs backend, administrateurs de base de données
3. **Équipe C : Intégration et Tests**
   - **Tâches** :
     - Intégration des services Firebase (authentification, notifications)
     - Tests unitaires et d'intégration
     - Tests utilisateurs et feedback
   - **Responsables** : Développeurs full-stack, testeurs QA

4 Ressources

4.1 Équipe de développement

1. **Chef de Projet**

   - Responsable de la planification, de la coordination et de la gestion globale du projet.
   - Assure la communication entre les différentes équipes et les parties prenantes.

2. **Développeurs Frontend**
   1. **Technologies utilisées** : Flutter pour le développement mobile, React.js ou Angular pour le développement web.
   2. **Responsabilités** :

      - Conception et développement de l'interface utilisateur.
      - Intégration des fonctionnalités frontend.
      - Collaboration avec les designers UX/UI pour garantir une expérience utilisateur fluide.

3. **Développeurs Backend**
    1. **Technologies utilisées** : Node.js avec Express.js.
    2. **Responsabilités** :
        - Développement et maintenance du serveur back-end.
        - Création et gestion des API REST.
        - Intégration de la base de données (Firestore, MongoDB).
4. **Spécialistes Firebase**
    1. **Responsabilités** :
        - Mise en place et gestion de l'authentification Firebase.
        - Intégration des notifications push.
        - Gestion du stockage des fichiers.
5. **Administrateurs de Base de Données**
    1. **Technologies utilisées** : Firestore, MongoDB.
    2. **Responsabilités** :
        - Conception et gestion de la base de données.
        - Optimisation des performances et de la sécurité des données.
6. **Testeurs QA (Assurance Qualité)**
    1. **Responsabilités** :
        - Tests unitaires et d'intégration.
        - Tests utilisateurs pour recueillir des feedbacks.
        - Identification et suivi des bugs.
7. **Designers UX/UI**
    1. **Responsabilités** :
        - Conception des maquettes et prototypes.
        - Collaboration avec les développeurs front-end pour l'implémentation de l'interface utilisateur.
        - Garantir une expérience utilisateur optimale et intuitive.

4.2 Outils et Technologies Utilisés

1. **Frontend Mobile**
    - **Technologie** : Flutter
        - **IDE** : Android studio ou Visual Studio Code
        - **Bibliothèques :** Provider, flutter Bloc
    - **Raison** : Framework open-source permettant de créer des applications mobiles nativement compilées avec une seule base de code.
2. **Frontend Web**
    - **Technologie** : React.js ou Angular

- **IDE** : Visual Studio Code, IntelliJ IDEA
- **Bibliothèques /Frameworks:**
  - React.js: Redux, React Router, Material-UI
  - Angular: NgRx, Angular Material

- **Raison** : Bibliothèques/frameworks populaires pour le développement d'interfaces utilisateur dynamiques et réactives.

3 **Backend**

- **Technologie** : Node.js avec Express.js
  - **IDE** : Visual Studio Code, IntelliJ IDEA
  - **Bibliothèques:** Mongoose (pour MongoDB), Firebase Admin SDK
- **Raison** : Plateforme JavaScript permettant de construire des applications back-end rapides et évolutives.

4 **Base de Données**

- **Technologies** : Firestore pour les données en temps réel, MongoDB pour des données flexibles
- **Raison** : Firestore offre une synchronisation en temps réel, tandis que MongoDB permet une flexibilité dans la structure des données.

5 **Services Supplémentaires**

- **Firebase** : Pour l'authentification, les notifications push, et le stockage des fichiers.
- **GitHub** : Pour le contrôle de version et la collaboration.
- **Jira ou Trello** : Pour la gestion des tâches et le suivi des progrès

6 **Contrôle de Version et Collaboration**

- **Outil** : GitHub
  - **Fonctionnalités** : Dépôts de code, gestion des versions, collaboration via pull requests

5 Gestion des Risques

5.1 Identification des risques

Risques technique

- **Incompatibilité des Capteurs** :
  - **Description** : Les capteurs intégrés (GPS, accéléromètre) peuvent ne pas être compatibles ou précis sur certains appareils.
  - **Impact** : Limite la précision et l'efficacité du suivi automatique des activités.
- **Problèmes de Performance** :

    - **Description** : L'application peut rencontrer des problèmes de performance, notamment des temps de réponse lents et des plantages.
    - **Impact** : Mauvaise expérience utilisateur, potentielle perte d'utilisateurs.
- **Intégration des Services Tiers** :
    - **Description** : Dépendance aux services Firebase pour l'authentification, les notifications et le stockage des fichiers.
    - **Impact** : Interruption ou modification des services tiers peut affecter le fonctionnement de l'application.

Risques de Sécurité

- **Violation de la Confidentialité des Données** :
    - **Description** : Les données personnelles et de santé des utilisateurs doivent être protégées contre les accès non autorisés.
    - **Impact** : Perte de confiance des utilisateurs, problèmes légaux.

Risques de Gestion de Projet

- **Dépassement des Délais** :
    - **Description** : Le projet peut ne pas être terminé dans les délais prévus en raison de diverses complications.
    - **Impact** : Retards dans le lancement, dépassement du budget.
- **Manque de Ressources** :
    - **Description** : Problèmes liés à la disponibilité ou à la compétence des ressources humaines nécessaires.
    - **Impact** : Retards dans le développement, qualité inférieure du produit final.

Risques de Marcher

- **Non-Adoption par les Utilisateurs** :
    - **Description** : L'application peut ne pas attirer suffisamment d'utilisateurs ou ne pas répondre à leurs besoins.
    - **Impact** : Faible taux de téléchargement, échec commercial de l'application.

5.2 Stratégies d'atténuation

Pour chaque risque identifié, des stratégies d'atténuation sont proposées pour minimiser l'impact et la probabilité de ces risques.

1. **Incompatibilité des Capteurs** :

    - **Stratégie** : Tester l'application sur une large gamme d'appareils pendant la phase de développement. Fournir des options de suivi manuel comme alternative.

2. **Problèmes de Performance** :

- **Stratégie** : Optimiser le code, effectuer des tests de charge et de stress. Utiliser des outils de surveillance de la performance en production.

3. **Intégration des Services Tiers** :

    - **Stratégie** : Avoir des plans de secours pour les services critiques. Effectuer des tests réguliers de la compatibilité des services.

4. **Violation de la Confidentialité des Données** :

    - **Stratégie** : Implémenter des mesures de sécurité robustes (chiffrement, authentification forte). Conformer à la réglementation sur la protection des données (GDPR, CCPA).

5. **Dépassement des Délais** :

    - **Stratégie** : Utiliser des méthodes de gestion de projet agile. Effectuer des revues régulières et ajuster les plans en conséquence.

6. **Manque de Ressources** :

    - **Stratégie** : Prévoir des ressources supplémentaires en cas de besoin. Former les membres de l'équipe pour qu'ils puissent assumer plusieurs rôles si nécessaires.

7. **Non-Adoption par les Utilisateurs** :

    - **Stratégie** : Effectuer des études de marché et des tests utilisateurs en amont. Recueillir et analyser les retours des utilisateurs pour améliorer l'application.

6 Budget

6.1 Estimation de cout

6.2 Répartition des budgets

7 Plan de Maintenance

Stratégie de Mise à Jour

- **Mises à Jour Régulières**
    - Mises à jour mensuelles pour améliorer les fonctionnalités et corriger les bugs.
- **Mises à Jour de Sécurité**
    - Patches de sécurité dès que des vulnérabilités sont identifiées.

- **Mises à Jour Majeures**
    - Nouvelles fonctionnalités et améliorations significatives, planifiées tous les six mois.

Gestion Continue

- **Suivi des Performances**
    - Surveillance continue des performances et de la disponibilité des services.
    - Utilisation d'outils d'analyse pour détecter et résoudre les problèmes de performance.
- **Support Client**
    - Support client disponible pour aider les utilisateurs avec les problèmes techniques et les questions.
- **Collecte de Feedback**
    - Recueillir des retours d'utilisateurs pour des améliorations continues.
- **Maintenance Préventive**
    - Audits réguliers du code et de la sécurité pour anticiper les problèmes potentiels.