**A. Identification**

**A.1. Identification du document**

| **Document:** |  |
| --- | --- |
| **Project:** | FITNESS-APP |
| **Version du projet:** | **1.0** |
| **Version du document:** | **1** |
| **Sécurité du document:** | CONFIDENTIEL |
| **Date de création:** | 1/07/2024 |
| **Par:** | LAINÉ STÉPHANE |

**A.2. Historique des changements**

| **Qui?** | **Quand?** | **Quoi?** |
| --- | --- | --- |
| **LAINE STEPHANE** |  |  |
|  |  |  |

# 

# 

# **Document de Projet : Application de Suivi de Fitness**

## **1. Introduction**

### **1.1 Contexte du Projet**

#### **Introduction**

Le mode de vie moderne est souvent caractérisé par une sédentarité croissante et des habitudes alimentaires peu saines. Ces facteurs contribuent à l'augmentation des problèmes de santé tels que l'obésité, les maladies cardiovasculaires et le diabète. Pour contrer ces tendances, il est essentiel de promouvoir des comportements plus actifs et sains. Cependant, beaucoup de personnes trouvent difficile de rester motivées et de suivre leurs progrès de manière efficace. C'est dans ce contexte que le projet d'application de suivi de fitness prend toute son importance.

#### **Problématique**

Les défis actuels liés à la santé et au bien-être comprennent :

* **Sédentarité** : Un nombre croissant de personnes passent de longues heures assises, que ce soit au travail ou à la maison.
* **Manque de motivation** : Sans des outils appropriés, il est difficile pour les individus de rester motivés et de poursuivre des objectifs de fitness sur le long terme.
* **Absence de suivi** : Beaucoup de personnes n'ont pas de moyen efficace de suivre leurs activités physiques et leurs progrès.
* **Accessibilité** : Les solutions existantes peuvent être coûteuses, complexes ou inaccessibles pour une grande partie de la population.

### **1.2 Objectifs du Projet**

Le projet de développement de l'application de suivi de fitness a pour principaux objectifs de fournir aux utilisateurs les outils nécessaires pour suivre, gérer et améliorer leur condition physique.

#### **1. Suivi des Activités Physiques**

#### **2. Définition et Suivi des Objectifs de Fitness**

#### **3. Visualisation des Statistiques et des Progrès**

#### **4. Notifications et Rappels**

#### **5. Interface Utilisateur Attrayante et Intuitive**

#### **6. Fonctionnalités Sociales (Optionnel)**

### **1.3 Portée du Projet**

La portée du projet délimite les fonctionnalités et services que l'application de suivi de fitness proposera ainsi que ceux qui ne sont pas inclus. Cette section précise ce que le projet englobe pour éviter toute confusion et aligner les attentes des parties prenantes.

#### **Fonctionnalités Incluses**

1. **Authentification et Gestion des Utilisateurs**
   * Inscription et connexion via email et mot de passe.
   * Authentification via des services tiers (Google, Facebook).
   * Gestion des profils utilisateurs (informations personnelles, préférences).
2. **Suivi des Activités Physiques**
   * Enregistrement manuel des séances d'entraînement (type d'exercice, durée, calories brûlées).
   * Suivi automatique via les capteurs de l'appareil (GPS pour la course, accéléromètre pour les pas).
   * Visualisation de l'historique des entraînements avec des filtres.
3. **Définition et Suivi des Objectifs de Fitness**
   * Définition d'objectifs personnalisés (perte de poids, distance de course, durée d'entraînement).
   * Suivi des progrès vers ces objectifs avec des mises à jour régulières et notifications.
   * Récompenses et badges pour l'atteinte des objectifs.
4. **Visualisation des Statistiques et des Progrès**
   * Tableaux de bord avec des graphiques pour visualiser les progrès quotidiens, hebdomadaires et mensuels.
   * Analyse des tendances et performances (meilleurs temps, augmentation des poids soulevés).
   * Comparaison des performances actuelles avec celles 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.

#### **Fonctionnalités Exclues**

1. **Conseils Nutritionnels**
   * L'application ne propose pas de plans ou conseils nutritionnels.
2. **Support Multilingue**
   * La première version de l'application ne prendra en charge que la langue principale du public cible (par exemple, l'anglais).
3. **Intégration avec des Appareils Externes**
   * L'intégration avec des appareils de fitness tiers (montres connectées, capteurs externes) ne sera pas incluse dans la version initiale.
4. **Suivi de la Santé Mentale**
   * L'application ne proposera pas de fonctionnalités spécifiques pour le suivi de la santé mentale (comme la méditation ou le suivi du sommeil).
5. **Programmes d'Entraînement Préétablis**
   * L'application ne fournira pas de programmes d'entraînement préétablis mais permettra aux utilisateurs de définir leurs propres objectifs et suivre leurs progrès.
6. **Support en Direct et Assistance Personnalisée**
   * Aucune assistance personnalisée ou support en direct ne sera inclus dans la version initiale de l'application

### **1.4 Parties Prenantes**

Identifier les principales parties prenantes impliquées dans le projet (utilisateurs finaux, développeurs, gestionnaires de projet, etc.).

## **2. Exigences du Projet**

### **2.1 Exigences Fonctionnelles**

Lister les fonctionnalités clés que l'application doit inclure :

* Authentification et gestion des utilisateurs.
* Suivi des séances d'entraînement.
* Définition et suivi des objectifs de fitness.
* Visualisation des statistiques et des progrès.
* Notifications et rappels.
* Interface utilisateur attrayante et intuitive.
* Fonctionnalités sociales (optionnelles).

### **2.2 Exigences Non Fonctionnelles**

Décrire les exigences de performance, de sécurité, de maintenabilité, d'accessibilité, etc.

## **3. Architecture du Système**

### **3.1 Vue d'Ensemble de l'Architecture**

Fournir un schéma de l'architecture du système, incluant le frontend, le backend et la base de données.

### **3.2 Architecture Frontend**

#### **technologies et Frameworks Utilisés pour le Développement des Applications Mobile et Web**

Pour le développement des applications mobile et web de l'application de suivi de fitness, les technologies et frameworks suivants seront utilisés :

#### **1. Frameworks Frontend**

1. **React Native (pour l'application mobile)**
   * **Description** : React Native est un framework open-source développé par Facebook qui permet de créer des applications mobiles pour iOS et Android en utilisant JavaScript et React.
   * **Avantages** :
     + **Cross-Platform** : Développer une seule fois et déployer sur les deux principales plateformes mobiles (iOS et Android).
     + **Performance Native** : Fournit une expérience utilisateur proche des applications natives.
     + **Large Communauté** : Bénéficie d'une vaste communauté de développeurs et de nombreux plugins disponibles.
2. **React.js (pour l'application web)**
   * **Description** : React.js est une bibliothèque JavaScript développée par Facebook pour construire des interfaces utilisateur. Elle permet de créer des composants réutilisables.
   * **Avantages** :
     + **Composants Réutilisables** : Favorise la réutilisation du code et la modularité.
     + **Performances** : Utilise un DOM virtuel pour améliorer les performances.
     + **Écosystème Riche** : Compatible avec un large éventail d'outils et de bibliothèques.

#### **2. Langages de Programmation**

1. **JavaScript**
   * **Description** : Langage de programmation principal utilisé pour le développement des interfaces utilisateur avec React.js et React Native.
   * **Avantages** :
     + **Universalité** : Utilisé aussi bien pour le développement frontend que backend (Node.js).
     + **Large Communauté** : Nombreux frameworks, bibliothèques et outils disponibles.
2. **TypeScript**
   * **Description** : Superset de JavaScript qui ajoute la typage statique optionnelle, ce qui aide à détecter les erreurs au moment de la compilation.
   * **Avantages** :
     + **Typage Statique** : Réduit les bugs en détectant les erreurs plus tôt dans le processus de développement.
     + **Amélioration du Code** : Facilite la maintenance et la lisibilité du code.

#### **3. Styles et Interfaces Utilisateur**

1. **CSS-in-JS (Styled Components)**
   * **Description** : Styled Components est une bibliothèque pour React et React Native qui permet de créer des composants de style en utilisant JavaScript.
   * **Avantages** :
     + **Scoping des Styles** : Les styles sont liés aux composants, réduisant les conflits de style.
     + **Dynamisme** : Permet d'utiliser des variables et des fonctions pour créer des styles dynamiques.
2. **Material-UI**
   * **Description** : Material-UI est une bibliothèque de composants React qui implémente les directives de conception Material Design de Google.
   * **Avantages** :
     + **Cohérence Visuelle** : Offre une cohérence visuelle à travers l'application avec des composants pré-stylisés.
     + **Personnalisable** : Hautement personnalisable pour correspondre à l'identité visuelle de l'application.

#### **4. Outils de Développement**

1. **Expo (pour React Native)**
   * **Description** : Expo est une suite d'outils et de services pour le développement et le déploiement d'applications React Native.
   * **Avantages** :
     + **Facilité d'Utilisation** : Simplifie le processus de développement React Native.
     + **Déploiement Rapide** : Permet de tester et de déployer rapidement des applications.
2. **Webpack (pour React.js)**
   * **Description** : Webpack est un module bundler pour JavaScript qui gère les dépendances et optimise les fichiers pour le déploiement.
   * **Avantages** :
     + **Optimisation** : Améliore les performances en optimisant les ressources.
     + **Modularité** : Permet de gérer les modules et les dépendances de manière efficace.

### **3.3 Architecture Backend**

#### **1. Technologies Utilisées**

1. **Node.js**
   * **Description** : Node.js est un environnement d'exécution JavaScript côté serveur. Il est non-bloquant et événementiel, ce qui le rend idéal pour les applications nécessitant une forte scalabilité.
   * **Avantages** :
     + **Performance** : Capable de gérer un grand nombre de connexions simultanées avec une faible latence.
     + **Écosystème** : Large collection de modules et de packages disponibles via npm.
2. **Express.js**
   * **Description** : Express.js est un framework web minimaliste pour Node.js. Il simplifie la création de serveurs web et d'API.
   * **Avantages** :
     + **Simplicité** : Facilite la gestion des routes et des middlewares.
     + **Flexibilité** : Hautement configurable et extensible.
3. **MongoDB**
   * **Description** : MongoDB est une base de données NoSQL orientée documents, idéale pour stocker des données structurées et semi-structurées.
   * **Avantages** :
     + **Scalabilité** : Facile à étendre horizontalement.
     + **Flexibilité** : Modèle de données flexible qui s'adapte bien aux changements.

#### **2. Structure Modulaire**

1. **Routes**
   * **Description** : Les routes définissent les points d'entrée de l'API. Elles mappent les URL aux contrôleurs correspondants.
   * **Exemple** :
     + /api/users : Gère les opérations liées aux utilisateurs.
     + /api/workouts : Gère les opérations liées aux séances d'entraînement.
2. **Contrôleurs**
   * **Description** : Les contrôleurs contiennent la logique métier pour traiter les requêtes entrantes, interagir avec les services et renvoyer les réponses appropriées.
   * **Exemple** :
     + UserController : Contient les fonctions pour l'inscription, la connexion, et la gestion des profils utilisateurs.
     + WorkoutController : Contient les fonctions pour créer, lire, mettre à jour et supprimer des séances d'entraînement.
3. **Services**
   * **Description** : Les services encapsulent la logique métier et les interactions avec les bases de données ou les API externes.
   * **Exemple** :
     + UserService : Gère les opérations sur les données des utilisateurs.
     + WorkoutService : Gère les opérations sur les données des séances d'entraînement.
4. **Models**
   * **Description** : Les modèles définissent la structure des documents de la base de données en utilisant Mongoose.
   * **Exemple** :
     + UserModel : Définit le schéma pour les utilisateurs.
     + WorkoutModel : Définit le schéma pour les séances d'entraînement.
5. **Middlewares**
   * **Description** : Les middlewares sont des fonctions qui s'exécutent avant les contrôleurs. Ils sont utilisés pour des tâches comme l'authentification et la validation des données.
   * **Exemple** :
     + authMiddleware : Vérifie si l'utilisateur est authentifié.
     + validationMiddleware : Valide les données des requêtes avant de les passer aux contrôleurs.

#### **3. API REST**

1. **Gestion des Utilisateurs**
   * **POST** /api/users/register : Inscription d'un nouvel utilisateur.
   * **POST** /api/users/login : Connexion d'un utilisateur.
   * **GET** /api/users/profile : Récupération du profil utilisateur (nécessite une authentification).
2. **Gestion des Séances d'Entraînement**
   * **POST** /api/workouts : Création d'une nouvelle séance d'entraînement.
   * **GET** /api/workouts : Récupération des séances d'entraînement de l'utilisateur.
   * **GET** /api/workouts/:id : Récupération d'une séance d'entraînement spécifique.
   * **PUT** /api/workouts/:id : Mise à jour d'une séance d'entraînement existante.
   * **DELETE** /api/workouts/:id : Suppression d'une séance d'entraînement.
3. **Intégration avec ExerciseDB**
   * **GET** /api/exercises : Récupération des exercices disponibles depuis ExerciseDB via RapidAPI.
   * **GET** /api/exercises/:id : Récupération des détails d'un exercice spécifique depuis ExerciseDB.

#### **4. Sécurité et Authentification**

1. **JWT (JSON Web Tokens)**
   * **Description** : JWT est utilisé pour l'authentification et l'autorisation. Les tokens sont générés lors de la connexion et utilisés pour sécuriser les routes.
   * **Avantages** :
     + **Sécurisé** : Les tokens sont signés et peuvent être vérifiés pour l'intégrité et l'authenticité.
     + **Stateless** : Permet une gestion des sessions sans état, facilitant l'évolutivité.
2. **Hashage des Mots de Passe**
   * **Description** : Les mots de passe des utilisateurs sont hashés avant d'être stockés dans la base de données pour garantir la sécurité.
   * **Outils** : bcrypt pour le hashage et la comparaison des mots de passe.

### **3.4 Base de Données**

Expliquer la structure de la base de données, y compris les types de données stockées et les technologies utilisées (Firestore, MongoDB).

## **4. Plan de Développement**

### **4.1 Phases du Projet**

#### **1. Phase 1 : Analyse**

**Objectif :** Comprendre les besoins des utilisateurs et définir les exigences fonctionnelles et non fonctionnelles de l'application.

* **Études de Marché et Analyse des Concurrents** : Analyse des applications de fitness existantes pour identifier les fonctionnalités clés et les lacunes potentielles.
* **Recueil des Besoins Utilisateurs** : Interroger les utilisateurs potentiels pour comprendre leurs attentes, leurs besoins et leurs préférences.
* **Spécifications Fonctionnelles** : Rédaction des spécifications détaillées des fonctionnalités requises, incluant la gestion des utilisateurs, le suivi des activités, la définition des objectifs, etc.
* **Spécifications Techniques** : Définition des exigences techniques, y compris les performances, la sécurité, l'évolutivité, et les intégrations nécessaires avec des services tiers comme ExerciseDB.

**Livrables :**

* Document de spécifications fonctionnelles et techniques.
* Analyse des risques et plan de mitigation.

#### **2. Phase 2 : Conception**

**Objectif :** Concevoir l'architecture technique et l'interface utilisateur de l'application.

* **Conception de l'Architecture** : Élaboration de l'architecture backend et frontend, incluant le choix des technologies et la structure des données.
* **Design de l'Interface Utilisateur (UI/UX)** : Création de wireframes et prototypes pour l'interface utilisateur, assurant une expérience utilisateur intuitive et agréable.
* **Modélisation de la Base de Données** : Conception du schéma de la base de données en fonction des besoins de l'application.
* **Planification du Développement** : Établissement du calendrier de développement et répartition des tâches entre les membres de l'équipe.

**Livrables :**

* Architecture technique détaillée.
* Prototypes UI/UX.
* Schéma de la base de données.
* Plan de projet détaillé.

#### **3. Phase 3 : Développement**

**Objectif :** Développer les différentes composantes de l'application en respectant les spécifications définies.

* **Développement Backend** : Mise en place du serveur backend, des API REST, et de l'intégration avec la base de données MongoDB et ExerciseDB.
* **Développement Frontend** : Développement des interfaces utilisateur pour les applications mobile et web en utilisant React Native et React.js.
* **Intégration Continue** : Mise en place d'un pipeline d'intégration continue pour automatiser les tests et les déploiements.
* **Documentation** : Rédaction de la documentation technique et utilisateur.

**Livrables :**

* Code source des applications backend et frontend.
* Documentation technique.
* Pipeline d'intégration continue.

#### **4. Phase 4 : Test**

**Objectif :** Valider la qualité, la performance et la sécurité de l'application avant son déploiement.

* **Tests Unitaires** : Test des composants individuels pour assurer leur bon fonctionnement.
* **Tests d'Intégration** : Vérification que les différentes composantes fonctionnent correctement ensemble.
* **Tests de Performance** : Mesure des performances de l'application sous différentes charges pour identifier et résoudre les goulots d'étranglement.
* **Tests de Sécurité** : Audit de sécurité pour s'assurer que l'application est protégée contre les vulnérabilités courantes.
* **Tests Utilisateurs** : Réalisation de tests utilisateurs pour recueillir des retours sur l'expérience utilisateur et effectuer les ajustements nécessaires.

**Livrables :**

* Rapport de tests avec les résultats et les actions correctives.
* Version bêta de l'application.

#### **5. Phase 5 : Déploiement**

**Objectif :** Lancer l'application sur les plateformes cibles et assurer son bon fonctionnement en production.

* **Déploiement Initial** : Publication de l'application sur les stores (App Store, Google Play) et mise en ligne de la version web.
* **Configuration de l'Infrastructure** : Mise en place des serveurs de production, des bases de données et des services cloud nécessaires pour le bon fonctionnement de l'application.
* **Monitoring et Support** : Surveillance continue de l'application en production pour identifier et résoudre rapidement tout problème.
* **Formation et Documentation** : Formation des utilisateurs finaux (si nécessaire) et distribution de la documentation utilisateur.

**Livrables :**

* Application déployée et opérationnelle.
* Infrastructure de production en place.
* Documentation utilisateur.
* Plan de support et maintenance.

.

### **4.2 Calendrier du Projet**

Fournir un calendrier avec les dates de début et de fin pour chaque phase du projet.

### **4.3 Ressources Nécessaires**

Identifier les ressources humaines, matérielles et financières nécessaires pour chaque phase du projet.

## **5. Gestion des Risques**

### **5.1 Identification des Risques**

Lister les principaux risques potentiels et leur impact sur le projet.

### **5.2 Plan de Mitigation des Risques**

Décrire les stratégies de mitigation pour chaque risque identifié.

## **6. Stratégie de Test**

### **6.1 Plan de Test**

Définir les types de tests à réaliser (unitaires, d'intégration, de performance) et leur calendrier.

### **6.2 Critères d'Acceptation**

Fixer les critères d'acceptation pour chaque fonctionnalité.

## **7. Déploiement et Maintenance**

### **7.1 Plan de Déploiement**

Expliquer les étapes nécessaires pour déployer l'application sur les plateformes ciblées.

### **7.2 Stratégie de Maintenance**

Décrire les plans pour la maintenance continue, les mises à jour et le support technique.

## **8. Documentation**

### **8.1 Documentation Utilisateur**

Prévoir la création de manuels et guides d'utilisation pour les utilisateurs finaux.

### **8.2 Documentation Technique**

Détailler la documentation nécessaire pour les développeurs et les équipes techniques.

## **9. Annexes**

### **9.1 Glossaire**

Définir les termes techniques et spécifiques utilisés dans le projet.

### **9.2 Références**

Lister toutes les références utilisées pour la conception et le développement du projet.