FITNESS-APP

.

A.1. Identification du document

Projet :	FITNESS-APP
Version du projet :	1.0
Version du document :	1
Sécurité du document :	Confidentiel
Date de création :	31/06/2024
Par :	LAINÉ STÉPHANE

A.2 HISTORIQUE DES CHANGEMENTS

A.2. Historique des changements

Qui?	Quand?	Quoi?
LAINE STEPHANE	25 juillet 2024	Création du document initial
LAINE STEPHANE	1er août 2024	Ajout de la section sur les besoins fonctionnels
LAINE STEPHANE	5 août 2024	Ajout des diagrammes UML
LAINE STEPHANE	6 août 2024	ajout des différentes phases du projet
LAINE STEPHANE	10 août 2024	Finalisation du document et correction des fautes

Document d'Analyse et de Conception pour l'Application de Suivi de Fitness

1. Introduction

1.1 Contexte du Projet

Avec l'augmentation de la conscience envers la santé et le bien-être, de nombreuses personnes cherchent des moyens efficaces pour suivre leur activité physique et atteindre leurs objectifs de fitness. L'application de suivi de fitness répond à ce besoin en offrant une solution intégrée qui permet aux utilisateurs de suivre leurs activités physiques, de définir des objectifs, de visualiser leurs progrès et de recevoir des notifications motivantes.

1.2 Objectifs du Projet

Les principaux objectifs du projet sont les suivants :

- 1. Suivi des activités physiques :

  L'application permet aux utilisateurs d'enregistrer et de suivre leurs séances d'entraînement de manière détaillée. Les utilisateurs peuvent ajouter manuellement des informations sur leurs séances, comme le type d'exercice, la durée, la distance parcourue, les calories brûlées, et d'autres données spécifiques à l'activité. Pour rendre le suivi encore plus pratique, l'application peut être intégrée à des dispositifs de suivi d'activité physique tels que les montres connectées ou les bracelets de fitness, permettant ainsi une collecte automatique des données. Les utilisateurs peuvent consulter l'historique de leurs séances, analyser leur performance sur une période donnée, et recevoir des résumés hebdomadaires ou mensuels pour évaluer leur progression.

- 2. Définition des objectifs :

L'application offre aux utilisateurs la possibilité de définir des objectifs de fitness personnalisés qui correspondent à leurs besoins et à leurs ambitions. Les utilisateurs peuvent fixer des objectifs à court terme, comme réaliser un certain nombre de pas quotidiens, ou des objectifs à long terme, comme perdre du poids ou augmenter leur endurance. Chaque objectif peut être spécifié en termes de temps, de distance, de fréquence ou de calories à brûler. L'application propose également des recommandations basées sur les données précédemment enregistrées, encourageant les utilisateurs à fixer des objectifs réalistes et atteignables. Une fois les objectifs définis, les utilisateurs peuvent suivre leur progression en temps réel et ajuster leurs plans en fonction des résultats obtenus.

- 3. Visualisation des progrès :
- L'application fournit des outils de visualisation des progrès réalisés, grâce à des graphiques interactifs et des tableaux de bord personnalisés. Les utilisateurs peuvent voir l'évolution de leurs performances au fil du temps, comparer leurs résultats avec les objectifs fixés, et identifier les tendances ou les domaines nécessitant une amélioration. Les graphiques peuvent être personnalisés pour afficher différentes mesures, comme la distance parcourue, le nombre de calories brûlées, ou le temps passé à s'entraîner. Les utilisateurs peuvent également comparer leurs performances sur différentes périodes, comme la comparaison des progrès réalisés au cours des mois ou des semaines précédents. Ces visualisations aident à maintenir la motivation et permettent une compréhension claire de l'impact des efforts fournis.
- 4. Notifications et rappels :

  Pour aider les utilisateurs à rester motivés et à atteindre leurs objectifs, l'application envoie des notifications et des rappels réguliers. Les utilisateurs peuvent configurer des rappels pour effectuer leurs séances d'entraînement, rester actifs tout au long de la journée, ou atteindre un objectif spécifique. Les notifications peuvent être personnalisées en fonction des préférences de chaque utilisateur, que ce soit en termes de fréquence, de contenu ou de moment d'envoi. Par exemple, un utilisateur peut choisir de recevoir une notification tous les matins pour l'encourager à commencer sa journée par une activité physique, ou un rappel en fin de journée pour enregistrer son activité. L'application peut également envoyer des messages de motivation, des conseils personnalisés, et des alertes lorsque l'utilisateur est proche d'atteindre un objectif ou lorsqu'il a besoin de redoubler d'efforts pour rester sur la bonne voie. Ces notifications jouent un rôle crucial dans l'engagement à long terme des utilisateurs.

1.3 Portée du Projet

Le projet inclut le développement des fonctionnalités suivantes :

- 1. Application mobile pour Android et iOS :

  L'application sera développée pour les plateformes Android et iOS, offrant une expérience utilisateur fluide et cohérente sur les deux systèmes d'exploitation. Elle sera optimisée pour fonctionner efficacement sur une large gamme d'appareils, garantissant ainsi une accessibilité maximale.

- 2. Application web accessible via les navigateurs modernes :

  En plus de l'application mobile, une version web sera disponible, accessible depuis tous les navigateurs modernes. Cela permettra aux utilisateurs de suivre leurs progrès et de gérer leurs activités depuis n'importe quel appareil doté d'un navigateur, que ce soit un ordinateur de bureau, une tablette, ou un smartphone.

  3. Suivi des activités avec intégration d'APIs pour l'enregistrement des données :

  Le suivi des activités sera amélioré grâce à l'intégration d'APIs qui permettront l'enregistrement automatique des données depuis des dispositifs connectés ou d'autres applications de fitness. Cela rendra la collecte des données plus précise et réduira la charge sur les utilisateurs en automatisant le processus.

- 4. Gestion des utilisateurs incluant l'authentification via des services tiers :

  L'application prendra en charge la gestion des utilisateurs avec une authentification sécurisée, y compris la possibilité de se connecter via des services tiers comme Google, Facebook, ou Apple. Cela simplifiera l'inscription et la connexion, tout en garantissant une sécurité renforcée.

- 5. Notifications push pour maintenir l'engagement des utilisateurs :

- Des notifications push seront utilisées pour garder les utilisateurs engagés en leur envoyant des rappels, des encouragements, ou des informations importantes directement sur leurs appareils. Ces notifications pourront être personnalisées en fonction des préférences de l'utilisateur et joueront un rôle clé dans la motivation continue.
-

Les fonctionnalités suivantes ne sont pas incluses dans la portée initiale :

- Fonctionnalités sociales: comme le partage d'activités sur les réseaux sociaux.
- Intégration avec des appareils de fitness spécifiques comme les montres connectées.

2. Analyse des Exigences

2.1 Exigences Fonctionnelles

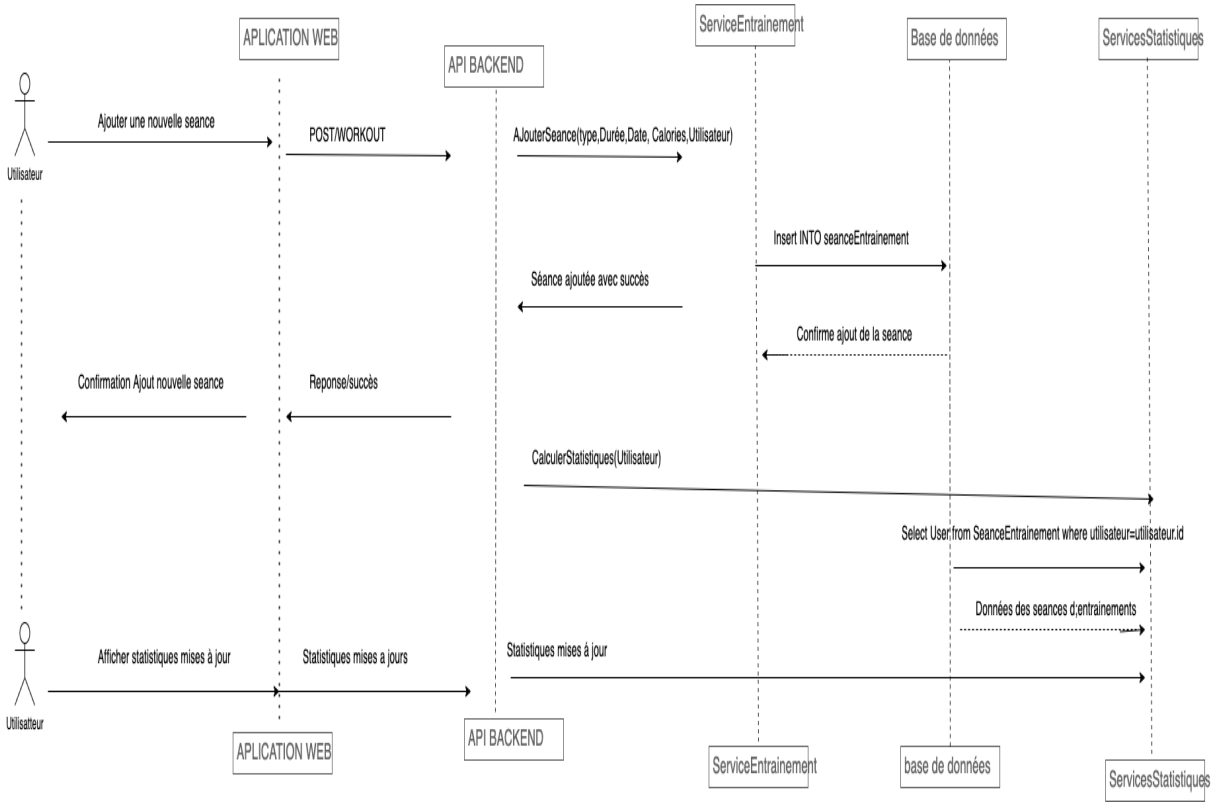
- **Enregistrement d'activités** : L'application doit permettre l'enregistrement des activités physiques telles que la course, la marche, le vélo, etc.
- **Définition d'objectifs** : Les utilisateurs doivent pouvoir définir des objectifs quotidiens, hebdomadaires ou mensuels.
- **Visualisation des progrès** : L'application doit fournir des tableaux de bord montrant les progrès réalisés.
- **Notifications** : L'application doit envoyer des notifications pour rappeler aux utilisateurs leurs objectifs et les encourager à rester actifs.

2.2 Exigences Non-Fonctionnelles

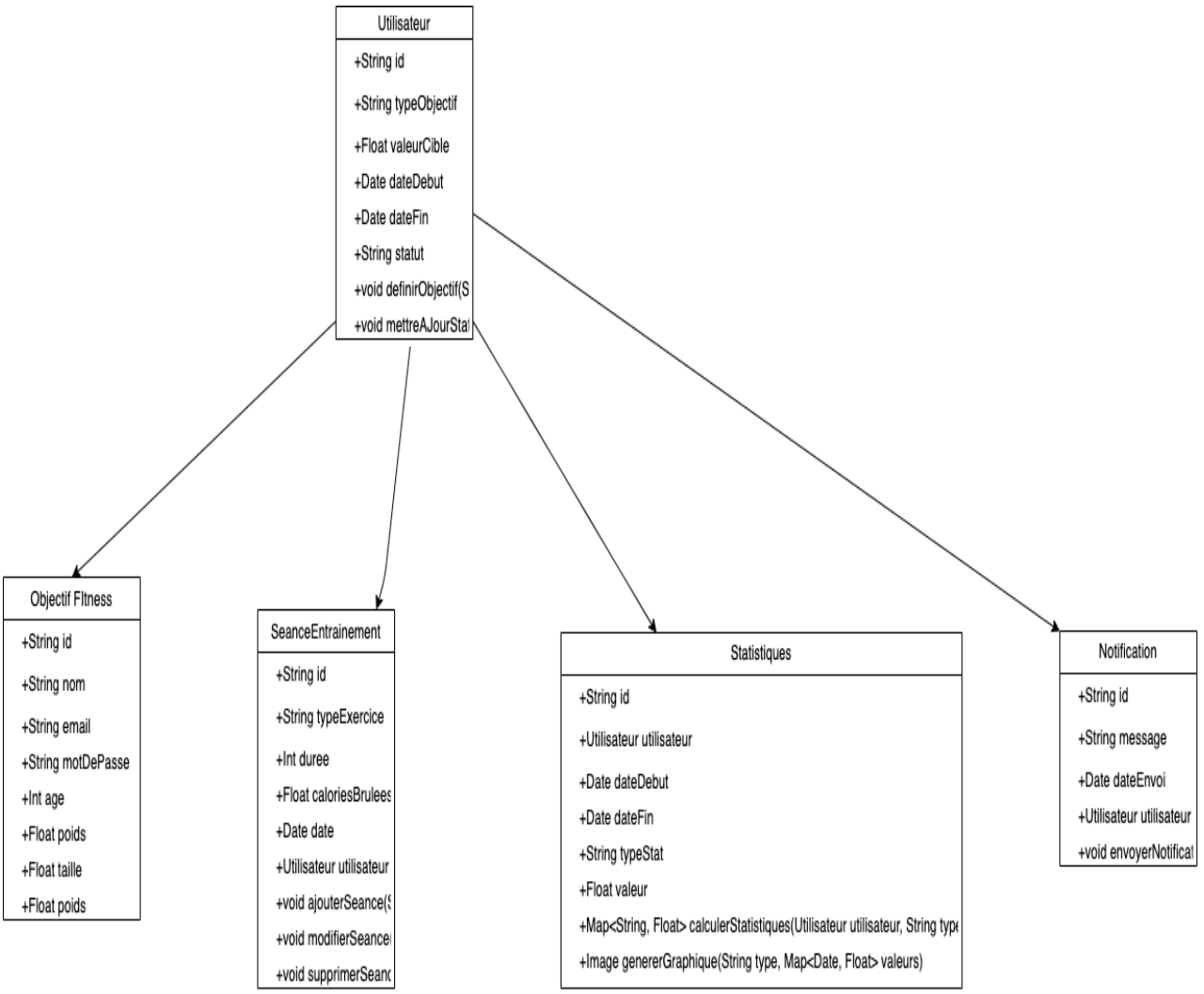
- **Performance** : L'application doit être capable de gérer jusqu'à 500 utilisateurs simultanés avec des temps de réponse inférieurs à 2 secondes.
- **Sécurité** : Les données des utilisateurs doivent être protégées contre les accès non autorisés et les attaques courantes.
- **Compatibilité** : L'application doit être compatible avec les versions récentes des systèmes d'exploitation Android et iOS, ainsi qu'avec les navigateurs web modernes.

2.3 Diagrammes UML

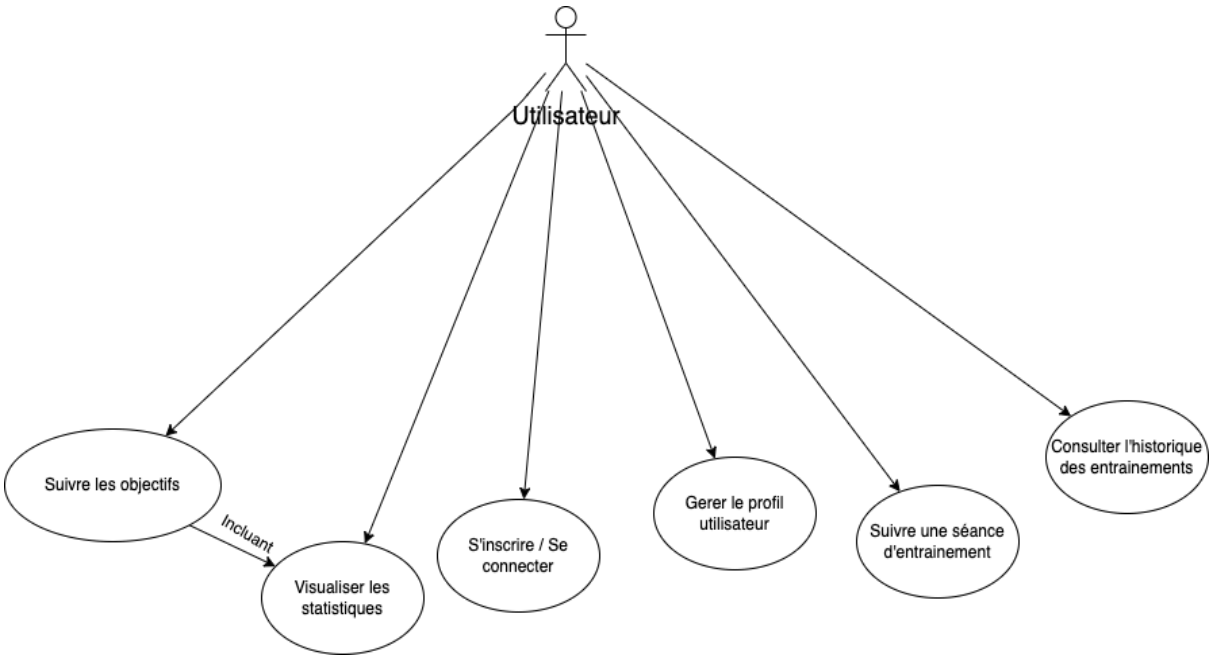
- **Diagramme de cas d'utilisation** : les interactions principales entre les utilisateurs et le système.



- **Diagramme de classes** : les principales classes du système, leurs attributs et leurs méthodes.



- **Diagramme de séquence** : Décrivez le flux des interactions entre les objets pour une fonctionnalité donnée.



3. Conception du Système

3.1 Architecture Globale

L'architecture de l'application est basée sur une architecture client-serveur, où le frontend est séparé du backend via des API REST. Le frontend inclut les applications mobile et web, tandis que le backend gère la logique métier, la gestion des utilisateurs, et l'intégration avec les services tiers.

3.2 Architecture Frontend

Technologies et Frameworks Utilisés

- **1. Mobile :**
- L'application mobile sera développée en utilisant React Native, ce qui permettra de créer une solution multiplateforme compatible à la fois avec Android et iOS. Cette approche garantira une expérience utilisateur cohérente et performante sur les deux systèmes d'exploitation, tout en optimisant le temps et les ressources de développement.
- **2. Web :**
- L'interface utilisateur web sera développée avec React.js, ce qui offrira une expérience utilisateur fluide et réactive. L'utilisation de React.js

permettra de créer des interfaces dynamiques et interactives, assurant une navigation intuitive et rapide pour les utilisateurs, quel que soit le navigateur ou l'appareil utilisé.

- **3. APIs :**
- L'application intégrera ExerciseDB via RapidAPI pour récupérer des données sur les exercices et les activités physiques. Cette intégration permettra d'accéder à une vaste base de données d'exercices, enrichissant l'expérience utilisateur avec des informations précises et variées, adaptées aux besoins de chaque utilisateur.
-

3.3 Architecture Backend

Technologies et API REST

- **Serveur Backend** : Développé en **Node.js** avec **Express.js** pour gérer les requêtes HTTP et les interactions avec la base de données.
- **Base de données** : Utilisation de **MongoDB** pour le stockage des données utilisateurs, des activités et des objectifs.
- **Authentification** : Intégration avec **Firebase Authentication** pour la gestion des utilisateurs et l'authentification via des services tiers (Google, Facebook).
- **API REST** : Mise en place d'APIs RESTful pour permettre la communication entre le frontend et le backend, incluant des endpoints pour la gestion des activités, des objectifs, et des utilisateurs.

4. Phases du Projet

4.1 Phase d'Analyse

Analyse des Parties Prenantes

Les parties prenantes (stakeholders) sont les individus ou groupes ayant un intérêt direct ou indirect dans le projet. Pour ce projet, les principales parties prenantes sont :

- Utilisateur final (Athlètes, Amateurs de fitness) : Ils utiliseront l'application pour suivre leurs activités physiques, définir des objectifs, et visualiser leurs progrès.
- Développeurs : L'équipe responsable du développement, de la maintenance, et des mises à jour de l'application.

- Gestionnaires de Projet : Responsables de la planification, de l'exécution et de la livraison du projet selon les délais et le budget.
- Experts en UX/UI : Chargés de concevoir une interface utilisateur attrayante et intuitive.
- Administrateurs système : Responsables de la gestion des serveurs et des bases de données.

2.2. Analyse des Besoins Utilisateurs

Pour comprendre les besoins des utilisateurs, nous pouvons créer des personas, qui sont des représentations fictives des utilisateurs types basées sur des recherches réelles.

- Persona 1 : Sarah, 28 ans, Passionnée de course à pied
    - Objectif : Suivre ses courses, voir ses progrès, et se fixer de nouveaux objectifs (ex. courir un marathon).
    - Besoins :
        - Interface simple pour entrer les données d'entraînement.
        - Graphiques clairs pour visualiser les progrès.
        - Notifications pour les rappels de séances et les récompenses.
        - Intégration avec des dispositifs de suivi comme une montre GPS.
- Persona 2 : Marc, 35 ans, Nouveau dans le fitness
    - Objectif : Perdre du poids et améliorer sa condition physique.
    - Besoins :
        - Guide sur les exercices adaptés à ses objectifs.
        - Suivi de ses progrès (perte de poids, amélioration des performances).
        - Encouragements sous forme de notifications ou de récompenses.

2.3. Identification des Cas d'Utilisation

Les cas d'utilisation sont les scénarios qui décrivent les interactions des utilisateurs avec le système pour atteindre un objectif spécifique. Voici quelques exemples de cas d'utilisation :

1. Inscription et Connexion

- Acteur Principal : Utilisateur
- Objectif : Permettre à un utilisateur de créer un compte et de se connecter à l'application.
- Scénario Principal :
  - L'utilisateur ouvre l'application.
  - Il sélectionne l'option "Créer un compte".
  - Il entre ses informations (nom, email, mot de passe).
  - Le système vérifie l'email et crée le compte.
  - L'utilisateur reçoit un email de confirmation et se connecte.
- Extensions :
  - Mot de passe oublié : L'utilisateur peut récupérer son mot de passe via email.

2. Enregistrement d'une Séance d'Entraînement
   - Acteur Principal : Utilisateur
   - Objectif : Enregistrer les détails d'une séance d'entraînement dans l'application.
   - Scénario Principal :
     - L'utilisateur sélectionne l'option "Ajouter une séance".
     - Il choisit le type d'exercice (course, musculation, etc.).
     - Il entre les détails (durée, distance, calories brûlées).
     - Le système enregistre la séance et met à jour l'historique d'entraînement de l'utilisateur.
   - Extensions :
     - Suivi automatique via GPS pour les courses en extérieur.
3. Définition et Suivi d'un Objectif de Fitness
   - Acteur Principal : Utilisateur
   - Objectif : Définir un objectif de fitness et suivre les progrès réalisés pour l'atteindre.
   - Scénario Principal :
     - L'utilisateur sélectionne l'option "Définir un objectif".
     - Il choisit un objectif (ex. courir 5 km en moins de 30 minutes).
     - Le système enregistre l'objectif et le suit lors des prochaines séances d'entraînement.
     - L'utilisateur reçoit des notifications sur ses progrès et des rappels pour l'encourager.

2.4. Besoins Fonctionnels

Les besoins fonctionnels décrivent ce que le système doit accomplir. Pour ce projet, voici les besoins fonctionnels identifiés :

- Gestion des utilisateurs
    - Inscription, connexion, déconnexion.
    - Gestion des profils utilisateurs (modification des informations personnelles, changement de mot de passe).
- Suivi des séances d'entraînement
    - Enregistrement manuel des séances (type d'exercice, durée, calories brûlées).
    - Suivi automatique des exercices via les capteurs de l'appareil (GPS, accéléromètre).
    - Historique des séances avec filtres par date, type d'exercice, etc.
- Objectifs de fitness
    - Définition des objectifs personnels.
    - Suivi des progrès avec des notifications de rappel.
    - Récompenses et badges pour les objectifs atteints.
- Visualisation des statistiques
    - Tableaux de bord avec graphiques pour suivre les progrès.
    - Comparaison des performances sur différentes périodes.
    - Analyse des tendances et des performances.
- 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.

2.5. Besoins Non Fonctionnels

Les besoins non fonctionnels définissent les critères qui jugeront le fonctionnement du système, plutôt que les comportements spécifiques.

- Performance
    - Temps de réponse rapide pour toutes les actions critiques (connexion, enregistrement de séance, etc.).
    - Gestion efficace des données en temps réel avec Firestore.
- Sécurité
    - Chiffrement des mots de passe (par exemple, avec bcrypt).

    - Protection des données personnelles en conformité avec le RGPD.
    - Authentification sécurisée via JWT (JSON Web Tokens).
- Scalabilité
    - Le système doit pouvoir supporter un grand nombre d'utilisateurs simultanés.
    - La base de données doit être optimisée pour gérer de grandes quantités de données historiques.
- Accessibilité
    - L'application doit être accessible sur différentes plateformes (iOS, Android, Web).
    - Compatibilité avec divers navigateurs et tailles d'écran.
- Fiabilité
    - Temps de disponibilité du système (SLA) : 99.9%.
    - Mise en place de sauvegardes régulières des données pour éviter toute perte.

2.6. Analyse des Contraintes Techniques

- Choix Technologiques
    - Frontend Mobile : Utilisation de Flutter pour une application cross-platform.
    - Frontend Web : Utilisation de React.js ou Angular pour une application web responsive.
    - Backend : Node.js avec Express.js pour un serveur léger et scalable.
    - Base de Données : Firestore pour les données en temps réel et MongoDB pour les données flexibles.
    - Services Supplémentaires : Firebase pour l'authentification, les notifications et le stockage de fichiers.
- Contraintes d'Intégration
    - Intégration avec des API tierces pour l'authentification (Google, Facebook).
    - Intégration avec des services de suivi d'activité physique (comme les capteurs GPS des smartphones).
- Contraintes de Déploiement
    - Hébergement sur une plateforme cloud (ex. AWS, Google Cloud).
    - Mise en place d'une CI/CD pour un déploiement continu et sans interruption.

2.7. Évaluation des Risques

- Risque 1 : Problèmes de performance liés au nombre d'utilisateurs croissant.
    - Stratégie : Utilisation de services cloud pour le scaling automatique, tests de charge réguliers.
- Risque 2 : Défaillance de la sécurité des données personnelles.
    - Stratégie : Implémentation de pratiques de sécurité de pointe, audits de sécurité réguliers.
- Risque 3 : Incompatibilité avec certaines versions de systèmes d'exploitation ou navigateurs.
    - Stratégie : Tests d'assurance qualité (QA) sur une large gamme d'appareils et de navigateurs.

4.2 Phase de Conception

. Architecture du Système

L'architecture du système définit la structure globale du projet, y compris les composants principaux et leurs interactions. Pour ce projet, nous utiliserons une architecture en couches qui sépare les différentes responsabilités :

1. Frontend (Côté Client)
    - Application Mobile : Développée avec Flutter pour assurer la compatibilité sur iOS et Android. Cette application permet aux utilisateurs de saisir des données, de suivre leurs progrès, et de recevoir des notifications.
    - Application Web : Développée avec React.js ou Angular pour une interface utilisateur réactive et dynamique accessible depuis n'importe quel navigateur. Cette application permet les mêmes fonctionnalités que l'application mobile mais avec une expérience optimisée pour le bureau.
2. Backend (Côté Serveur)
    - API REST : Construite avec Node.js et Express.js pour gérer les requêtes provenant des applications frontend. Cette API gère la logique métier, la validation des données, et la communication avec les bases de données.
    - Services d'Authentification : Gérés par Firebase Authentication pour faciliter la gestion des utilisateurs (inscription, connexion, réinitialisation de mot de passe, etc.) via des fournisseurs OAuth comme Google et Facebook.
3. Base de Données

    - Firestore : Utilisé pour stocker les données en temps réel, telles que les séances d'entraînement, les objectifs, et les statistiques utilisateur. Firestore permet un accès rapide et une synchronisation en temps réel avec l'application.
    - MongoDB : Utilisé pour stocker des données structurées ou semi-structurées nécessitant une gestion complexe, comme les historiques détaillés d'entraînement, les configurations utilisateur, ou les métadonnées des fichiers.
4. Infrastructure Cloud
    - Hébergement : Utilisation de services cloud (comme AWS ou Google Cloud) pour héberger les serveurs backend et les bases de données. Le choix d'une plateforme cloud permet de bénéficier d'une scalabilité automatique, de la sécurité, et d'un déploiement continu.
    - CI/CD : Intégration continue et déploiement continu configurés via des outils comme GitHub Actions, Jenkins, ou CircleCI pour assurer des mises à jour régulières et fiables du système.

4.2. Diagrammes UML

4.2.1. Diagramme de Cas d'Utilisation

Le diagramme de cas d'utilisation illustre les différentes interactions entre les utilisateurs (ou acteurs) et le système.

- Acteurs Principaux :
    - Utilisateur : Accède à l'application, enregistre des séances d'entraînement, définit des objectifs.
    - Administrateur : Gère les utilisateurs, modifie les configurations du système.
- Cas d'Utilisation :
    - Inscription et Connexion
    - Enregistrement de Séance d'Entraînement
    - Définition d'Objectifs
    - Visualisation des Statistiques
    - Gestion des Notifications

4.2.2. Diagramme de Séquence

Le diagramme de séquence montre le déroulement des interactions entre les différents composants du système pour un cas d'utilisation spécifique.

- Cas d'Utilisation : Enregistrement d'une Séance d'Entraînement
    1. L'utilisateur sélectionne l'option "Ajouter une séance".
    2. L'application envoie une requête POST à l'API REST.
    3. Le serveur valide les données et les enregistre dans Firestore.
    4. Le serveur renvoie une réponse de succès à l'application.
    5. L'utilisateur voit la séance ajoutée dans son historique.

4.2.3. Diagramme de Classes

Le diagramme de classes montre la structure des classes et leurs relations dans le système.

- Classes Principales :
    - Utilisateur : Attributs comme nom, email, mot de passe, et méthodes comme s'inscrire, se connecter.
    - Séance : Attributs comme type d'exercice, durée, calories, et méthodes comme enregistrer une séance, calculer les calories.
    - Objectif : Attributs comme type d'objectif, date de début/fin, et méthodes comme définir un objectif, suivre un objectif.
    - Notification : Attributs comme type de notification, message, date d'envoi, et méthodes comme envoyer une notification, gérer les préférences.

4.2.4. Diagramme de Déploiement

Le diagramme de déploiement décrit l'architecture physique du système, notamment comment les différents composants logiciels sont déployés sur le matériel.

- Serveur Backend : Hébergé sur une plateforme cloud (AWS/Google Cloud) avec un serveur Node.js pour gérer les requêtes API.
- Base de Données : Firestore pour les données en temps réel et MongoDB pour les données complexes, tous deux hébergés sur le cloud.
- Clients : Application mobile déployée sur iOS et Android via les stores respectifs. Application web déployée sur un serveur web et accessible via un navigateur.

4.3. Modélisation des Données

4.3.1. Schéma de la Base de Données

- Collections Firestore :
    - Users : Contient les profils utilisateurs, avec des documents représentant chaque utilisateur.
    - Workouts : Contient les séances d'entraînement, avec des documents représentant chaque séance, liés à un utilisateur spécifique.
    - Goals : Contient les objectifs de fitness définis par les utilisateurs, avec des documents pour chaque objectif.
- Tables MongoDB :
    - UserSettings : Stocke les configurations utilisateur, telles que les préférences de notification et les informations de profil détaillées.
    - WorkoutHistory : Stocke l'historique détaillé des séances, avec des informations comme les zones de fréquence cardiaque, les segments GPS pour les courses, etc.

4.3.2. API REST

- Endpoints Principaux :
    - POST /users/register : Créer un nouveau compte utilisateur.

    - POST /users/login : Authentification de l'utilisateur.
    - GET /workouts : Récupérer l'historique des séances d'entraînement de l'utilisateur.
    - POST /workouts : Enregistrer une nouvelle séance d'entraînement.
    - GET /goals : Récupérer les objectifs de l'utilisateur.
    - POST /goals : Définir un nouvel objectif.
- Gestion des Erreurs :
    - 400 Bad Request : Pour les données mal formées ou invalides.
    - 401 Unauthorized : Pour les tentatives d'accès non authentifiées.
    - 500 Internal Server Error : Pour les erreurs inattendues côté serveur.

4.4. Conception de l'Interface Utilisateur (UI/UX)

4.4.1. Wireframes

- Écran de Connexion : Affiche des champs pour l'email et le mot de passe, avec un bouton de connexion et une option pour récupérer le mot de passe.
- Tableau de Bord : Montre un aperçu des statistiques récentes, les progrès vers les objectifs, et les notifications récentes.

4.4.2. Prototypes Haute-Fidélité

- Design Mobile : Présentation des écrans clés comme le suivi en temps réel d'une course, la vue des progrès hebdomadaires, et l'interface de saisie de nouvelles séances.
- Design Web : Disposition optimisée pour les écrans plus larges, avec une vue d'ensemble du tableau de bord, des graphiques interactifs, et des filtres pour analyser les données.

4.4.3. Design System

- Palette de Couleurs : Choix de couleurs pour refléter un sentiment de dynamisme et de bien-être, tout en assurant une bonne lisibilité (. vert pour les progrès, rouge pour les alertes).
- Typographie : Utilisation de polices modernes et claires pour faciliter la lecture et la navigation.
- Composants UI : Boutons, champs de saisie, cartes de données, menus, tous définis avec un guide de styles cohérent.

4.3 Phase de Développement

- Développement du frontend pour les applications mobile et web.
- Développement du backend, incluant les APIs REST et l'intégration avec les services tiers.
- Tests unitaires et d'intégration pour valider le bon fonctionnement de chaque composant.

4.4 Phase de Test

- Tests fonctionnels pour valider que chaque fonctionnalité répond aux exigences.
- Tests de performance et de sécurité pour s'assurer de la robustesse du système.

4.5 Phase de Déploiement

- Déploiement de l'application sur les stores mobiles (Google Play, Apple App Store) et sur le web.
- Mise en place de la surveillance et des logs pour détecter et résoudre rapidement les problèmes post-déploiement.

5. Conclusion

Ce document fournit une vue d'ensemble du projet de développement de l'application de suivi de fitness, incluant l'analyse des exigences, la conception du système, et la planification des phases du projet. Ce cadre servira de référence tout au long du cycle de vie du projet pour assurer que les objectifs sont atteints de manière efficace et que l'application répond aux attentes des utilisateurs.