REPUBLIQUE DU CAMEROUN
****
PAIX-TRAVAIL-PATRIE
****
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
****
UNIVERSITE DE MAROUA
****
FACULTE DE SCIENCE
****
DEPARTEMENT DE MATHEMATIQUES ET INFORMATIQUE
****

REPUBLIC OF CAMEROON
****
PEACE-WORK-FATHERLAND
****
MINISTRY OF HIGHER EDUCATION
****
UNIVERSITY OF MAROUA
******
FACULTY OF SCIENCES
****
DEPARTMENT OF MATHEMATICAL AND COMPUTER SCIENCE
****

INFORMATIQUE NIVEAU 3

TPE : GENIE LOGICIEL 2

CODE :INF325

Thème : Gestion du Don de Sang à Maroua

NOMS	PRENOMS	MATRICULES
ARSENE	DJIBANGAR	21A0002FS
BOKOUANOUDJIEL	VIVIANE	22A0799FS
DJIDE BENY	BERTRAND	22A0650FS
FANEZOUNE	DANIEL	22A0297FS
GENEVIEVE MAIBAYANG	TEMOUA	23FS0129
ROBALBAYE GAYE	ROMUALD	22A0264FS

Chargé du cours : Mr BAYANG SOULOUKNA

ANNEE ACCADEMIQUE

2025-2026

PLAN DU TRAVAIL

INTRODUCTION

Le présent document constitue le cahier des charges de l'application mobile B-life, une solution innovante de gestion et de coordination des dons de sang dans la ville de Maroua. Ce projet vise à répondre aux besoins urgents en matière de transfusion sanguine en connectant rapidement les receveurs avec les donneurs compatibles et disponibles. L'application B-life permettra de sauver des vies en réduisant considérablement le temps nécessaire pour trouver un donneur de sang compatible lors de situations d'urgence médicale.

I OBJECTIFS DU PROJET

1 Présentation du projet

B-life est une application mobile destinée à faciliter l'accès au don de sang dans la région de Maroua. Face aux urgences médicales nécessitant des transfusions sanguines, il est souvent difficile de localiser rapidement des donneurs compatibles et disponibles. L'application met en relation trois types d'utilisateurs :

Les clients/receveurs : personnes recherchant du sang en urgence ;

Les donneurs : personnes volontaires pour donner leur sang ;

Les administrateurs : personnels médicaux gérant la validation des donneurs.

Le système intègre une géolocalisation, un système de notification en temps réel et une gestion automatisée des disponibilités basée sur les délais réglementaires entre deux dons.

2. Objectifs du projet

a) Objectif général

Créer une plateforme mobile efficace permettant de connecter rapidement les personnes nécessitant du sang avec des donneurs compatibles et disponibles à Maroua.

b) Objectifs spécifiques

- Constituer une base de données fiable de donneurs de sang testés et validés ;
- Permettre aux clients de rechercher du sang selon leur groupe sanguin en cas d'urgence ;
- Localiser automatiquement le donneur compatible le plus proche et disponible ;
- Respecter le délai réglementaire de 3 mois entre deux dons de sang ;
- Notifier rapidement les donneurs potentiels et obtenir leur confirmation ;
- Orienter les utilisateurs vers le complexe sanitaire le plus proche ;
- Assurer un suivi transparent des demandes et des dons effectués.

II. DESCRIPTION DU SYSTÈME

1. Fonctionnalités

A. Fonctionnalités pour les Clients/Receveurs

- Inscription et authentification
  - ✓ Création de compte avec informations personnelles;
  - ✓ Renseignement du groupe sanguin;
  - ✓ Indication de l'âge, ville et quartier;
  - ✓ Géolocalisation automatique ou manuelle;
  - ✓ Connexion sécurisée ;
  - ✓ Nombre de poche de sang.
- Demande de sang
  - ✓ Recherche urgente de sang par groupe sanguin ;
  - ✓ Indication du niveau d'urgence ;
  - ✓ Localisation automatique du client.

- Notification en temps réel du statut de la demande
  - ✓ Réception des informations du complexe sanitaire Suivi ;
  - ✓ Consultation de l'historique des demandes ;
  - ✓ Notifications de mise à jour ;
  - ✓ Évaluation du service.

B. Fonctionnalités pour les Donneurs

- Inscription et profil
  - ✓ Enregistrement par l'administrateur après tests médicaux ;
  - ✓ Consultation du profil personnel ;
  - ✓ Visualisation du statut de disponibilité ;
  - ✓ Historique des dons effectués.
- Gestion des demandes
  - ✓ Réception de notifications push lors de demandes compatibles ;
  - ✓ Confirmation ou refus de disponibilité dans un délai de 2 minutes ;
  - ✓ Réception des détails du complexe sanitaire en cas de confirmation ;
  - ✓ Calcul automatique de l'éligibilité (délai de 3 mois).
- Historique et statistiques
  - ✓ Consultation des dons passés ;
  - ✓ Nombre de vies sauvées ;
  - ✓ Date du prochain don possible.

C. Fonctionnalités pour les Administrateurs

- Gestion des donneurs
  - ✓ Ajout de nouveaux donneurs après tests médicaux
  - ✓ Validation des résultats des tests sanguins
  - ✓ Désactivation ou suppression de donneurs
  - ✓ Consultation de la base de données complète
- Gestion des demandes
  - ✓ Suivi en temps réel des demandes
  - ✓ Intervention manuelle si nécessaire - Statistiques globales
- Gestion des complexes sanitaires
  - ✓ Ajout et modification des centres de santé
- Géolocalisation des établissements
  - ✓ Mise à jour des informations de contact
- Tableaux de bord et rapports
  - ✓ Statistiques sur les dons effectués
  - ✓ Taux de réponse des donneurs
  - ✓ Analyse géographique des demandes
  - ✓ Groupes sanguins les plus demandés

D. Fonctionnalités du Système Automatisé

Algorithme de recherche intelligente

- ✓ Filtrage par groupe sanguin compatible
- ✓ Vérification automatique de l'éligibilité (3 mois minimum depuis le dernier don)
- ✓ Tri des donneurs par proximité géographique
- ✓ Élargissement progressif de la zone de recherche

Système de notification

- ✓ Envoi de notifications push prioritaires
- ✓ Gestion du délai de réponse de 2 minutes
- ✓ Notification automatique du donneur suivant en cas de non-réponse
- ✓ Notification au client du statut de sa demande

Géolocalisation

- ✓ Calcul de distance en temps réel
- ✓ Recherche du complexe sanitaire le plus proche - Optimisation des trajets

2. Contraintes techniques et non techniques

A. Contraintes techniques

Plateforme et compatibilité

- ✓ Application mobile native ou hybride (iOS et Android)
- ✓ Interface responsive adaptée à différentes tailles d'écran
- ✓ Fonctionnement optimal sur Android 8.0+ et iOS 12+
- ✓ Connexion internet requise pour les fonctionnalités principales
- ✓ Mode hors ligne limité pour consultation de profil

Performance

- ✓ Temps de réponse du système inférieur à 3 secondes
- ✓ Notification en temps réel (latence < 5 secondes)

Sécurité et confidentialité

- ✓ Chiffrement des données sensibles (SSL/TLS)
- ✓ Authentification sécurisée avec hash des mots de passe
- ✓ Conformité RGPD pour la protection des données personnelles
- ✓ Système de permissions pour l'accès aux données de localisation
- ✓ Sauvegarde régulière des données

Technologies recommandées

- ✓ Frontend : React Native ou Flutter
- ✓ Backend : Node.js avec Express ou Django
- ✓ Base de données : PostgreSQL ou MongoDB
- ✓ Service de géolocalisation : Google Maps API ou OpenStreetMap
- ✓ Notifications push : Firebase Cloud Messaging (FCM)
- ✓ Hébergement : AWS, Google Cloud ou Azure

B. Contraintes non techniques

Contraintes réglementaires et médicales

- ✓ Respect du délai minimum de 3 mois entre deux dons de sang
- ✓ Validation médicale obligatoire avant inscription d'un donneur
- ✓ Tests sanguins conformes aux normes sanitaires
- ✓ Âge minimum et maximum pour les donneurs selon la réglementation camerounaise
- ✓ Confidentialité des informations médicales

Contraintes organisationnelles

- ✓ Partenariat avec les centres de santé de Maroua
- ✓ Formation des administrateurs à l'utilisation de la plateforme
- ✓ Sensibilisation de la population au don de sang
- ✓ Disponibilité 24h/24 et 7j/7 pour les urgences

Contraintes d'utilisabilité

- ✓ Interface intuitive et facile à utiliser en situation d'urgence
- ✓ Support multilingue (français, anglais)

Contraintes géographiques

- ✓ Couverture initiale limitée à Maroua
- ✓ Extension progressive à d'autres villes du Cameroun
- ✓ Adaptation au réseau mobile local

III. PHASE DU PROJET

1. Planification du projet

Phase 1 : Analyse et conception (4 Jours)

Jour 1-2 : Analyse des besoins

- ✓ Étude de marché et analyse de la concurrence
- ✓ Rencontre avec les parties prenantes (centres de santé, donneurs potentiels)
- ✓ Validation du cahier des charges
- ✓ Définition des spécifications fonctionnelles détaillées

Jour 3-4 : Conception

- ✓ Création des maquettes UI/UX (wireframes et prototypes)
- ✓ Conception de l'architecture système
- ✓ Modélisation de la base de données
- ✓ Élaboration des diagrammes UML
- ✓ Validation de la conception avec le client

Phase 2 : Développement (4 semaines)

Jour 5-6 : Setup et infrastructure

- ✓ Configuration de l'environnement de développement
- ✓ Mise en place du serveur backend
- ✓ Configuration de la base de données
- ✓ Intégration des services tiers (maps, notifications)

Jour 7-9 : Développement du backend

- ✓ Développement des API REST
- ✓ Implémentation de la logique métier
- ✓ Système d'authentification et autorisation
- ✓ Algorithme de recherche de donneurs
- ✓ Service de géolocalisation
- ✓ Système de notifications

Jour 10-13 : Développement du frontend

- ✓ Interface d'inscription et connexion
- ✓ Tableau de bord client/receveur
- ✓ Tableau de bord donneur

- ✓ Panel administrateur
- ✓ Intégration des cartes et géolocalisation
- ✓ Système de notifications push

Jour 14 : Intégration

- ✓ Connexion frontend-backend
- ✓ Tests d'intégration
- ✓ Optimisation des performances

Phase 3 : Tests et validation (3 semaines)

Jour 15-16 : Tests

- ✓ Tests unitaires de chaque module
- ✓ Tests d'intégration
- ✓ Tests de sécurité
- ✓ Tests de charge et performance
- ✓ Tests d'ergonomie
- ✓ Tests sur différents appareils

Jour 17 : Validation

- ✓ Tests utilisateurs avec groupe pilote
- ✓ Corrections de bugs
- ✓ Validation finale par le client
- ✓ Préparation de la documentation

Phase 4 : Déploiement et formation (2 semaines)

Jour 18 : Déploiement

- ✓ Configuration des serveurs de production
- ✓ Déploiement sur Play Store et App Store
- ✓ Configuration des systèmes de monitoring - Mise en place des sauvegardes

Jour 19 : Formation et lancement

- ✓ Formation des administrateurs
- ✓ Campagne de sensibilisation
- ✓ Lancement officiel
- ✓ Support utilisateurs

Phase 5 : Maintenance et amélioration continue

- ✓ Support technique continu
- ✓ Correction de bugs
- ✓ Mises à jour régulières
- ✓ Ajout de nouvelles fonctionnalités basées sur les retours utilisateurs

- ✓ Monitoring et optimisation des performances

Livrables

- ✓ Code source complet (frontend et backend)
- ✓ Documentation technique
- ✓ Documentation utilisateur
- ✓ Maquettes et designs
- ✓ Rapports de tests
- ✓ Plan de maintenance

IV. ANALYSE ET CONCEPTION

1. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation illustre les interactions entre les différents acteurs (Client, Donneur, Administrateur) et le système B-life.

Acteurs principaux :

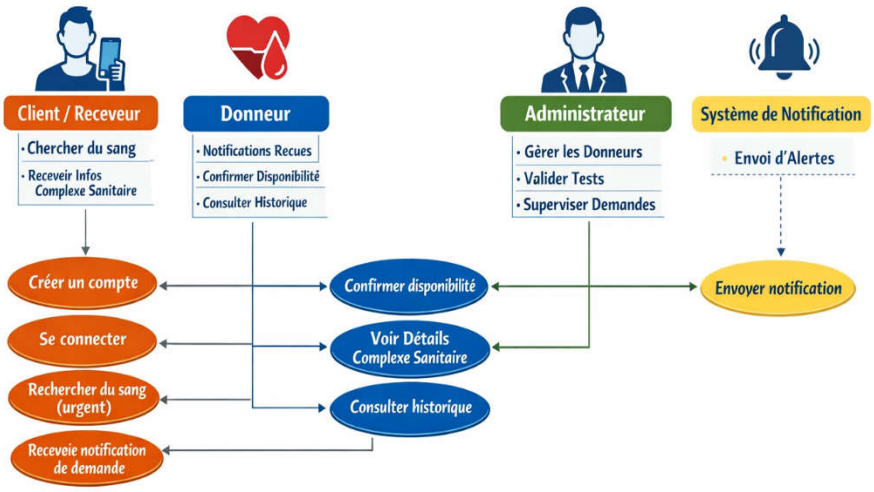
- ✓ Client/Receveur : Crée un compte, recherche du sang, reçoit les informations du complexe sanitaire
- ✓ Donneur : Reçoit des notifications, confirme sa disponibilité, consulte son historique
- ✓ Administrateur : Gère les donneurs, valide les tests, supervise les demandes
- ✓ Système de notification : Acteur secondaire gérant l'envoi automatique des alertes

Cas d'utilisation principaux :

- ✓ Créer un compte
- ✓ Se connecter
- ✓ Rechercher du sang (urgent)
- ✓ Recevoir notification de demande

Confirmer disponibilité

- ✓ Voir détails complexe sanitaire
- ✓ Gérer les donneurs
- ✓ Tester et valider les donneurs
- ✓ Consulter historique



2. Diagramme de classes

Le diagramme de classes représente la structure statique du système avec les principales entités et leurs relations.

Classes principales :

Utilisateur (classe abstraite)

- ✓ Attributs : id, nom, prénom, email, téléphone, motDePasse, role, dateInscription
- ✓ Méthodes : seConnecter(), seDeconnecter(), modifierProfil()

Client (hérite de Utilisateur)

Attributs spécifiques : groupeSanguin, âge, ville, quartier, localisation

- ✓ Méthodes : créerCompte(), rechercherSang(), voirHistorique()
- ✓ Donneur (hérite de Utilisateur)
- ✓ Attributs spécifiques : groupeSanguin, âge, ville, quartier, localisation, dernierDon, estValidé, estDisponible, dateTest
- ✓ Méthodes : confirmerDisponibilité(), consulterHistorique(), vérifierÉligibilité()

Administrateur (hérite de Utilisateur)

✓ Attributs spécifiques : niveau

✓ Méthodes : ajouterDonneur(), validerDonneur(), gérerDemandes(), consulterStatistiques()

Demande

✓ Attributs : id, groupeSanguin, urgence, dateDemande, statut, latitude, longitude - Méthodes : créerDemande(), annulerDemande(), mettreÀJourStatut()

Notification

✓ Attributs : id, dateEnvoi, message, estLue, typeNotification

✓ Méthodes : envoyerNotification(), marquerCommeLue()

ComplexeSanitaire

✓ Attributs : id, nom, adresse, ville, quartier, latitude, longitude, téléphone

✓ Méthodes : trouverPlusProche(), getDetails()

Don

✓ Attributs : id, dateDon, groupeSanguin, quantité, statut

✓ Méthodes : enregistrerDon(), consulterDetails()

Relations :

✓ Un Client effectue plusieurs Demandes (1 à n)

✓ Un Donneur réalise plusieurs Dons (1 à n)

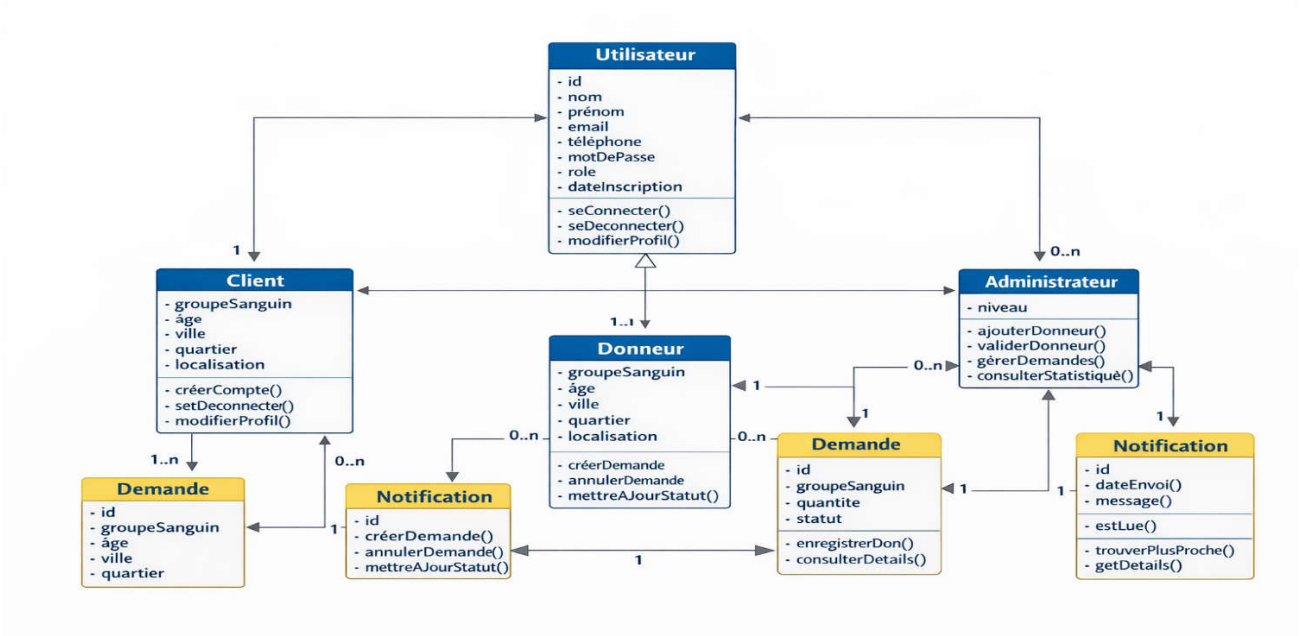
✓ Une Demande génère plusieurs Notifications (1 à n)

✓ Un Donneur reçoit plusieurs Notifications (1 à n)

✓ Une Demande se fait dans un ComplexeSanitaire (1 à 1)

✓ Un Administrateur gère plusieurs Donneurs (1 à n)

✓ Une Demande est satisfaite par un Donneur (1 à 1)



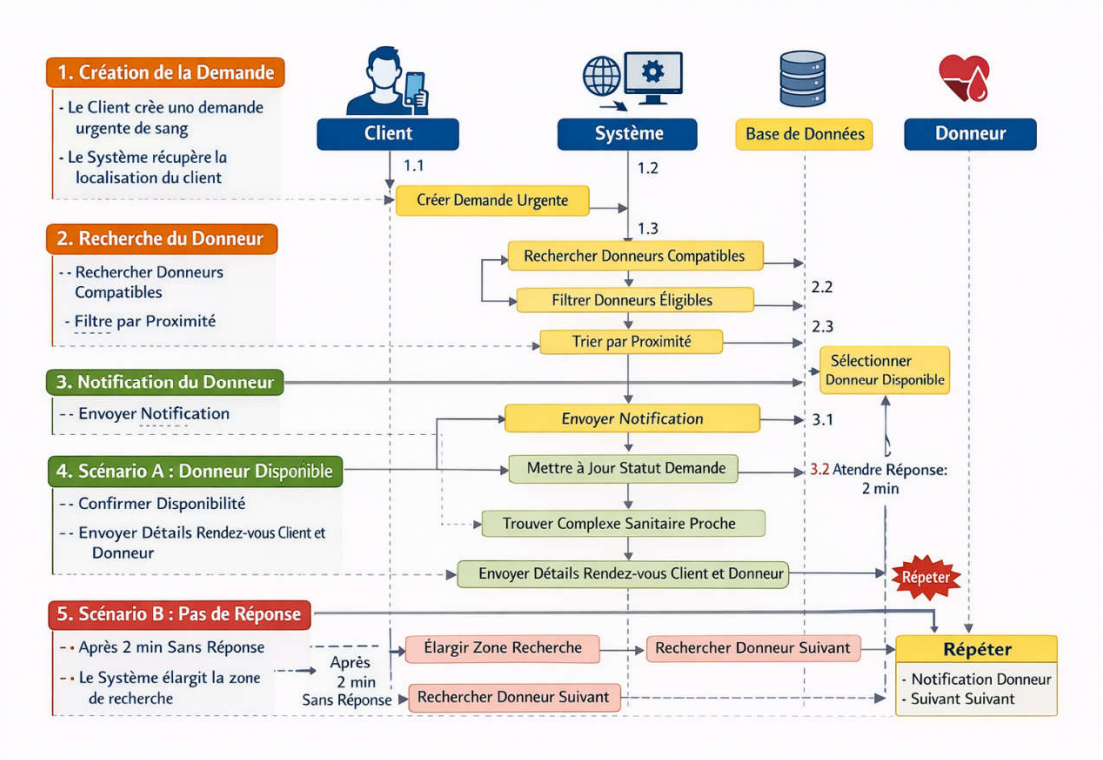
3. Diagramme de séquence

Le diagramme de séquence illustre le flux d'interactions lors d'une demande urgente de sang.

Scénario : Recherche et mise en relation pour don de sang

Étapes principales :

1. Création de la demande
   - ✓ Le Client crée une demande urgente de sang
   - ✓ Le Système récupère la localisation du client
   - ✓ La demande est enregistrée en base de données
2. Recherche du donneur
   - ✓ Le Système recherche les donneurs compatibles (groupe sanguin)
   - ✓ Filtre les donneurs éligibles (dernier don > 3 mois)
   - ✓ Tri par proximité géographique
   - ✓ Sélection du donneur le plus proche
3. Notification du donneur
   - ✓ Le Système envoie une notification au donneur sélectionné
   - ✓ Attente de réponse (temps : 2 minutes)
4. Scénario A : Donneur disponible
   - ✓ Le Donneur confirme sa disponibilité
   - ✓ Le Système met à jour le statut de la demande
   - ✓ Recherche du complexe sanitaire le plus proche
   - ✓ Envoi des détails du rendez-vous au Client et au Donneur
5. Scénario B : Pas de réponse
   - ✓ Après 2 minutes sans réponse
   - ✓ Le Système élargit la zone de recherche
   - ✓ Sélection du donneur suivant dans la liste
   - ✓ Notification du donneur suivant
   - ✓ Répétition du processus jusqu'à trouver un donneur disponible voir le diagramme interactif dans l'artéfact



V. IMPLÉMENTATION

1. Architecture technique

Architecture 3-tiers

- ✓ Couche présentation : Application mobile (React Native/Flutter)
- ✓ Couche métier : API REST (Node.js/Django)
- ✓ Couche données : Base de données (PostgreSQL/MongoDB)

2. API RESTE

Authentification

- ✓ POST /api/auth/register - Inscription
- ✓ POST /api/auth/login - Connexion
- ✓ POST /api/auth/logout - Déconnexion
- ✓ GET /api/auth/profile - Profil utilisateur

Clients

GET /api/clients/:id - Détails client

- ✓ PUT /api/clients/:id - Modifier profil
- ✓ GET /api/clients/:id/demandes - Historique demandes

Demandes

- ✓ POST /api/demandes - Créer demande urgente
- ✓ GET /api/demandes/:id - Détails demande

- ✓ PUT /api/demandes/:id/annuler - Annuler demande
- ✓ GET /api/demandes/:id/statut - Statut en temps réel

Donneurs

- ✓ GET /api/donneurs/:id - Détails donneur
- ✓ PUT /api/donneurs/:id/disponibilite - Confirmer/refuser
- ✓ GET /api/donneurs/:id/historique - Historique dons
- ✓ GET /api/donneurs/:id/eligibilite - Vérifier éligibilité

Administrateur

- ✓ POST /api/admin/donneurs - Ajouter donneur
- ✓ PUT /api/admin/donneurs/:id/valider - Valider donneur
- ✓ GET /api/admin/donneurs - Liste tous les donneurs - GET

/api/admin/statistiques - Statistiques globales

POST /api/admin/complexes - Ajouter complexe sanitaire

Recherche

- ✓ POST /api/recherche/donneurs - Rechercher donneurs compatibles
- ✓ GET /api/complexes/proche - Trouver complexe proche

Notifications

- ✓ POST /api/notifications/envoyer - Envoyer notification
- ✓ GET /api/notifications/:donneur_id - Notifications d'un donneur
- ✓ PUT /api/notifications/:id/lue - Marquer comme lue

3. Algorithme de recherche de donneur`

ALGORITHME RechercherDonneur(demande):

// Étape 1 : Filtrage initial

groupe_demande =

demande.groupe_sanguin

donneurs_compatibles

FILTRER(Donneurs) PAR:

- ✓ groupe_sanguin COMPATIBLE avec groupe_demande
- ✓ est_valide = TRUE
- ✓ (date_actuelle - dernier_don) >= 90 jours

SI donneurs_compatibles EST VIDE:RETOURNER "Aucun donneur éligible disponible"

Étape 2 : Tri par proximité

Pour chaque donneur DANS donneurs_compatibles: distance =
CalculerDistance(demande.localisation, donneur.localisation) donneur.distance = distance
donneurs_tries = TRIER(donneurs_compatibles) PAR distance CROISSANT

Étape 3 : Notification séquentielle

rayon_recherche = 5 km // Rayon initial
max_rayon = 50 km // Rayon maximum
timeout = 120 secondes // Délai d'attente

TANT QUE rayon_recherche <= max_rayon:

donneurs_dans_rayon = FILTRER(donneurs_tries) PAR distance <=
rayon_recherche

POUR chaque donneur DANS donneurs_dans_rayon:
// Envoyer notification
EnvoyerNotification(donneur, demande)

// Attendre réponse
reponse = AttendreReponse(donneur, timeout)

SI reponse == "DISPONIBLE":
complexe = TrouverComplexeProche(donneur, demande.client)
NotifierClient(demande.client, donneur, complexe)
NotifierDonneur(donneur, complexe)
EnregistrerMatch(demande, donneur, complexe)
RETOURNER "Donneur trouvé"

SINON SI reponse == "INDISPONIBLE":
CONTINUER // Passer au donneur suivant

SINON: // Pas de réponse (timeout)
CONTINUER

// Élargir la zone de recherche
rayon_recherche = rayon_recherche + 5 km

RETOURNER "Aucun donneur disponible après recherche élargie"

4. Technologies et frameworks recommandés

Frontend Mobile

- ✓ React Native avec Flutter
- ✓ Redux/MobX pour la gestion d'état
- ✓ React Navigation pour la navigation
- ✓ Axios pour les appels API
- ✓ React Native Maps pour la géolocalisation

Backend

- ✓ Node.js avec Express.js
- ✓ JWT pour l'authentification
- ✓ Socket.io pour les communications temps réel
- ✓ Nodemailer pour les emails

Base de données

- ✓ PostgreSQL avec PostGIS pour les données géospatiales
- ✓ Redis pour le cache et les sessions

Services tiers

Firebase Cloud Messaging pour les notifications push

- ✓ Google Maps API ou Mapbox pour la cartographie
- ✓ AWS S3 pour le stockage de fichiers
- ✓ Sentry pour le monitoring d'erreurs

DevOps

- ✓ Docker pour la conteneurisation - GitHub Actions ou GitLab CI/CD
- ✓ AWS/Google Cloud/Azure pour l'hébergement
- ✓ Nginx comme reverse proxy

VI. DÉVELOPPEMENT ET TESTS UNITAIRES

1. Méthodologie de développement

Approche Agile - Scrun

- ✓ Sprints de 2 semaines
- ✓ Daily stand-ups
- ✓ Sprint planning et retrospectives
- ✓ Revues de code systématiques

Bonnes pratiques

- ✓ Version control avec Git (GitFlow)
- ✓ Code review obligatoire avant merge
- ✓ Documentation du code

- ✓ Respect des conventions de nommage
- ✓ Architecture modulaire et réutilisable

2. Tests unitaires

Backend - Exemples de tests

Test de vérification d'éligibilité d'un donneur

```
TEST VerifierEligibiliteDonneur:
donneur = CreerDonneur(dernier_don = date_actuelle - 100 jours)
resultat = donneur.verifierEligibilite()
ASSERT resultat == TRUE

donneur2 = CreerDonneur(dernier_don = date_actuelle - 60 jours)
resultat2 = donneur2.verifierEligibilite()

ASSERT resultat2 == FALSE
```

Test de calcul de distance

```
TEST CalculerDistance:
point1 = {localisation receveur}  // Maroua
point2 = {localisation donneur}
distance = CalculerDistance(point1, point2)
ASSERT distance > 0 ET distance < 5  // Distance en km
```

Test de compatibilité groupe sanguin

```
TEST VerifierCompatibiliteGroupeSanguin:
ASSERT EstCompatible("O-", "A+") == TRUE
ASSERT EstCompatible("A+", "O-") == FALSE
ASSERT EstCompatible("AB+", "O+") == TRUE
ASSERT EstCompatible("O+", "AB-") == FALSE
```

Test d'authentification

```
TEST ConnexionUtilisateur:
utilisateur = CreerUtilisateur(email="test@blife.cm", password="Test123!")
token = Authentifier(email="test@blife.cm", password="Test123!")
ASSERT token != NULL

token_invalide = Authentifier(email="test@blife.cm", password="MauvaisPass")
ASSERT token_invalide == NULL
```

3. Tests d'intégration

Test de flux complet de demande

TEST FluxDemandeComplete:

- Création client

  client = CreerClient(groupe_sanguin="A+")

- **Création donneur éligible**

  donneur = CreerDonneur(groupe_sanguin="A+", dernier_don=date - 100 jours)

- **Création demande**

  demande = CreerDemande(client, groupe_sanguin="A+")

- **Recherche donneur**

  donneur_trouve = RechercherDonneur(demande)

  ASSERT donneur_trouve != NULL

- Notification envoyée

  notification = ObtenirDerniereNotification(donneur)

  ASSERT notification.demande_id == demande.id

- **Confirmation donneur**

  ConfirmerDisponibilite(donneur, demande)

- **Vérification statut**

  demande_mise_a_jour = ObtenirDemande(demande.id)

  ASSERT demande_mise_a_jour. statut == "DONNEUR_TROUVE"

4. Tests de performance

Tests de charge

- ✓ 100 utilisateurs simultanés créant des demandes
- ✓ Temps de réponse API < 3 secondes
- ✓ 1000 notifications envoyées en < 10 secondes

Tests de stress

- ✓ Limite du système sous charge extrême
- ✓ Comportement en cas de panne d'un service - Récupération après crash

5. Tests d'acceptation utilisateur (UAT)

Scénarios de test

Scénario 1 : Inscription et création de compte client

- ✓ L'utilisateur ouvre l'application
- ✓ Clique sur "Créer un compte"
- ✓ Remplit le formulaire avec ses informations
- ✓ Accepte la localisation
- ✓ Le compte est créé avec succès

Scénario 2 : Demande urgente de sang

- ✓ Le client se connecte
- ✓ Clique sur "Demande urgente"
- ✓ Sélectionne son groupe sanguin
- ✓ Confirme la demande
- ✓ Reçoit une notification de recherche en cours
- ✓ Reçoit les détails du complexe sanitaire dans les 5 minutes

Scénario 3 : Réponse du donneur à une notification

- ✓ Le donneur reçoit une notification push
- ✓ Ouvre la notification
- ✓ Lit les détails de la demande
- ✓ Confirme sa disponibilité
- ✓ Reçoit l'adresse du complexe sanitaire

6. Critères d'acceptation

Performance

- ✓ Temps de chargement de l'application < 3 secondes
- ✓ Notification reçue en moins de 5 secondes après envoi
- ✓ Recherche de donneur < 5 minutes

Fonctionnel

- ✓ 100% des demandes traitées
- ✓ Aucun donneur notifié avant 3 mois après son dernier don

- Géolocalisation précise à 100m près

Sécurité

- ✓ Données chiffrées en transit et au repos
- ✓ Authentification sécurisée (hash + salt)
- ✓ Pas de fuite de données personnelles

Utilisabilité

- ✓ Interface intuitive (90% des utilisateurs réussissent les tâches sans aide)
- ✓ Accessibilité (WCAG 2.1 niveau AA)
- ✓ Support multilingue fonctionne

ANNEXES

A. Glossaire

Don de sang : Acte volontaire de prélèvement de sang en vue d'une transfusion

Groupe sanguin : Classification du sang basée sur les antigènes présents (A, B, AB, O avec Rhésus+/-)

Éligibilité : Condition permettant à un donneur de donner son sang (délai de 3 mois respecté)

Complexe sanitaire : Établissement de santé où se déroule le don de sang

Géolocalisation : Détermination de la position géographique d'un utilisateur

Push notification : Notification mobile envoyée en temps réel

B. Compatibilité des groupes sanguins

Receveurs ⟶ Donneurs compatibles

- O- : peut recevoir de O- O+ : peut recevoir de O-, O+
- A- : peut recevoir de O-, A-
- A+ : peut recevoir de O-, O+, A-, A+
- B- : peut recevoir de O-, B-
- B+ : peut recevoir de O-, O+, B-, B+
- AB- : peut recevoir de O-, A-, B-, AB- AB+ : peut recevoir