

# CAHIER DES CHARGES

## PLATEFORME WEB AKAWA - ASSOCIATION CAMEROUNAISE DE KARATÉ WADOKAI

**Version :** 1.0

**Date :** Août 2025

**Client :** Association Camerounaise de Karaté Wadokai (AKAWA)

**Projet :** Site web vitrine promotionnel

---

### 1. PRÉSENTATION DU PROJET

#### 1.1 Contexte et Enjeux

Le karaté Shotokan domine actuellement la scène martiale camerounaise et africaine, éclipsant d'autres styles tout aussi riches et diversifiés. Le karaté Wadokai (Wadoyo), reconnu comme l'un des deuxièmes styles les plus répandus mondialement, mérite une visibilité accrue au Cameroun et en Afrique.

L'Association Camerounaise de Karaté Wadokai (AKAWA), dirigée par des maîtres reconnus comme Maître Rosic, Maître Kenfack Anissé, et Maître Aouna, souhaite développer une plateforme digitale moderne pour promouvoir ce style ancestral et fédérer sa communauté.

#### 1.2 Problématique

- Manque de visibilité du karaté Wadokai face à la dominance du Shotokan
- Absence de plateforme centralisée pour les clubs et pratiquants Wadokai
- Difficulté à valoriser les succès des karatékas camerounais à l'international
- Besoin d'attirer de nouveaux pratiquants et partenaires

#### 1.3 Objectifs Stratégiques

**Objectif Principal :** Créer une plateforme web moderne et interactive pour promouvoir le karaté Wadokai au Cameroun, en Afrique et dans le monde.

**Objectifs Spécifiques :**

- Répertorier et valoriser tous les clubs Wadokai enregistrés au Cameroun sous AKAWA
- Présenter l'histoire et l'évolution du karaté Wadokai
- Mettre en avant les succès des karatékas camerounais à l'international
- Faciliter le recrutement de nouveaux pratiquants

- Développer des partenariats internationaux
  - Créer une communauté digitale engagée
- 

## 2. SLOGAN ET POSITIONNEMENT

### 2.1 Slogan Principal

"WADOKAI CAMEROUN - L'Art Martial de l'Excellence"

### 2.2 Slogans Alternatifs

- "Tradition, Excellence, Fierté Camerounaise"
- "Du Cameroun au Monde, l'Esprit Wadokai"
- "Wadokai : La Voie de l'Harmonie et de la Force"

### 2.3 Positionnement

- **Premium** : Excellence technique et tradition authentique
  - **Inclusif** : Ouvert à tous, débutants comme experts
  - **International** : Rayonnement mondial depuis le Cameroun
  - **Innovant** : Tradition ancestrale avec approche moderne
- 

## 3. SPÉCIFICATIONS TECHNIQUES

### 3.1 Technologies Requises

**Architecture Full-Stack Modern :**

**Frontend :**

- React.js (version 18+) avec TypeScript
- Tailwind CSS pour le design system
- Framer Motion pour les animations
- Three.js pour les animations 3D
- React Router pour la navigation SPA
- TanStack Query pour la gestion d'état et cache
- React Hook Form pour la validation des formulaires

**Backend Serverless :**

- **Netlify Functions** (Node.js/TypeScript serverless)
- **Neon PostgreSQL** (base de données serverless auto-scale)
- **Prisma ORM** pour la gestion de base de données
- **JWT Authentication** avec refresh tokens
- **Zod** pour la validation des schémas API

#### Services Externes :

- **WhatsApp Business API** (237 675 39 52 38)
- **Cloudinary** pour la gestion des médias
- **Resend** pour les emails transactionnels
- **Stripe** pour les paiements (cotisations)

#### Architecture SOLID & MVC Full-Stack :

typescript

```
// Backend Models (Prisma Schema)
model User {
  id      String @id @default(cuid())
  email   String @unique
  name    String
  phone   String?
  clubId  String?
  club    Club? @relation(fields: [clubId], references: [id])
  createdAt DateTime @default(now())
}

model Club {
  id      String @id @default(cuid())
  name    String
  location String
  instructor String
  phone   String
  members User[]
}

// Netlify Functions Controllers
// netlify/functions/clubs.ts
export const handler: Handler = async (event, context) => {
  const clubController = new ClubController(
    new PrismaClubRepository()
  );





  return await clubController.handleRequest(event);
};
```

## Déploiement et Infrastructure :

- **Netlify** pour le frontend et fonctions serverless
- **Neon** pour PostgreSQL serverless avec branching
- **GitHub Actions** pour CI/CD avancé
- **Vercel Analytics** pour le monitoring
- Environment variables sécurisées

## 3.2 Design System et UI/UX

### Palette de Couleurs :

- Primaire : Rouge traditionnel ( #DC2626) - Force et passion
- Secondaire : Or/Jaune ( #F59E0B) - Excellence et prestige
- Neutre : Gris moderne ( #374151) - Professionnalisme
- Accent : Vert Cameroun ( #059669) - Identité nationale

### **Typographie :**

- Titre : Police moderne sans-serif (Inter, Poppins)
- Corps : Police lisible (Open Sans, Source Sans Pro)
- Accent : Police calligraphique pour les éléments traditionnels

### **Animations et Interactions :**

- Animations fluides avec Framer Motion
  - Transitions micro-interactives
  - Éléments 3D pour les démonstrations de techniques
  - Parallax scrolling pour l'immersion
  - Hover effects sophistiqués
- 

## **4. ARCHITECTURE ET FONCTIONNALITÉS**

### **4.1 Structure du Site**

#### **Page d'Accueil**

- Hero section avec animation 3D de karatéka
- Présentation AKAWA avec compteurs animés
- Témoignages de maîtres avec carrousel interactif
- Actualités récentes
- Appel à l'action pour rejoindre

#### **À Propos**

- Histoire du karaté Wadokai mondial
- Implantation au Cameroun
- Présentation des maîtres fondateurs
- Vision et mission AKAWA
- Timeline interactive de l'évolution

## **Clubs et Dojos**

- Carte interactive des clubs au Cameroun
- Fiches détaillées par club (adresse, horaires, contacts)
- Système de recherche par région
- Galeries photos des dojos
- Informations sur les instructeurs

## **Nos Champions**

- Galerie des karatékas camerounais à l'international
- Profils détaillés avec palmarès
- Vidéos de compétitions
- Témoignages et parcours inspirants
- Système de filtre par pays/compétition

## **Formations et Stages**

- Calendrier interactif des événements
- Descriptions détaillées des stages
- Système d'inscription en ligne
- Ressources pédagogiques téléchargeables
- Certifications et grades

## **Partenariats**

- Présentation des partenaires actuels
- Opportunités de collaboration
- Formulaire de contact partenaires
- Témoignages de partenaires
- Avantages du partenariat

## **Actualités et Blog**

- Articles sur le karaté Wadokai
- Résultats de compétitions
- Interviews de maîtres

- Conseils d'entraînement
- Événements communautaires

## **Contact et Adhésion**

- Formulaires de contact intelligents
- Processus d'adhésion simplifié
- Coordonnées des responsables
- FAQ interactive
- Support multilingue (français, anglais)

## **4.2 Architecture Full-Stack SOLID**

### **Structure Backend Serverless :**

typescript

```
// Database Schema (Prisma)
// prisma/schema.prisma
generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL") // Neon connection
}

model User {
  id      String @id @default(cuid())
  email   String @unique
  name    String
  phone   String?
  level   String? // Ceinture/Grade
  clubId  String?
  club    Club? @relation(fields: [clubId], references: [id])
  events  EventRegistration[]
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
}

model Club {
  id      String @id @default(cuid())
  name    String
  location String
  city    String
  instructor String
  phone   String
  email   String?
  latitude Float?
  longitude Float?
  members User[]
  events  Event[]
  isActive Boolean @default(true)
  createdAt DateTime @default(now())
}

model Event {
  id      String @id @default(cuid())
  title   String
```



```

description String?
date      DateTime
location  String
clubId    String?
club      Club? @relation(fields: [clubId], references: [id])
registrations EventRegistration[]
maxParticipants Int?
price     Float?
}

model EventRegistration {
  id      String @id @default(cuid())
  userId  String
  eventId String
  user    User   @relation(fields: [userId], references: [id])
  event   Event  @relation(fields: [eventId], references: [id])
  status  String @default("pending") // pending, confirmed, cancelled

  @@unique([userId, eventId])
}

model Partner {
  id      String @id @default(cuid())
  name    String
  description String?
  website String?
  logo    String?
  contact String
  type     String // sponsor, dojo, federation, etc.
  isActive Boolean @default(true)
}

```

## Netlify Functions API (Controllers) :

```
typescript
```

```
// netlify/functions/api/clubs.ts
import { Handler } from '@netlify/functions';
import { PrismaClient } from '@prisma/client';
import { ClubController } from '../../src/controllers/ClubController';
import { PrismaClubRepository } from '../../src/repositories/PrismaClubRepository';

const prisma = new PrismaClient();

export const handler: Handler = async (event, context) => {
  // CORS headers
  const headers = {
    'Access-Control-Allow-Origin': '*',
    'Access-Control-Allow-Headers': 'Content-Type, Authorization',
    'Access-Control-Allow-Methods': 'GET, POST, PUT, DELETE, OPTIONS',
  };

  if (event.httpMethod === 'OPTIONS') {
    return { statusCode: 200, headers, body: '' };
  }

  try {
    const clubRepository = new PrismaClubRepository(prisma);
    const clubController = new ClubController(clubRepository);

    const response = await clubController.handleRequest({
      method: event.httpMethod,
      path: event.path,
      body: event.body ? JSON.parse(event.body) : null,
      queryParams: event.queryStringParameters,
    });

    return {
      statusCode: response.status,
      headers: { ...headers, 'Content-Type': 'application/json' },
      body: JSON.stringify(response.data),
    };
  } catch (error) {
    console.error('API Error:', error);
    return {
      statusCode: 500,
      headers,
      body: JSON.stringify({ error: 'Internal Server Error' }),
    };
  }
}
```

```
}  
};
```

**Services Business Logic :**

typescript

```

// src/services/WhatsAppService.ts
export class WhatsAppService {
  private readonly phoneNumber = "237675395238";

  async sendNotification(type: NotificationType, data: NotificationData): Promise<void> {
    const message = this.formatMessage(type, data);

    // Envoi WhatsApp + sauvegarde en base
    await this.sendWhatsAppMessage(message);
    await this.logNotification(type, data, message);
  }

  private formatMessage(type: NotificationType, data: NotificationData): string {
    switch(type) {
      case 'NEW_MEMBER':
        return `👤 NOUVEAU MEMBRE AKAWA\n\n` +
          `Nom: ${data.name}\n` +
          `Club: ${data.clubName}\n` +
          `Téléphone: ${data.phone}\n` +
          `Email: ${data.email}\n` +
          `Niveau: ${data.level || 'Débutant'}\n\n` +
          `Merci de contacter ce nouveau membre ! 👥`;

      case 'EVENT_REGISTRATION':
        return `📅 INSCRIPTION ÉVÉNEMENT\n\n` +
          `Événement: ${data.eventTitle}\n` +
          `Participant: ${data.memberName}\n` +
          `Date: ${data.eventDate}\n` +
          `Contact: ${data.memberPhone}`;

      case 'PARTNERSHIP_REQUEST':
        return `💖 DEMANDE PARTENARIAT\n\n` +
          `Entreprise: ${data.companyName}\n` +
          `Contact: ${data.contactPerson}\n` +
          `Téléphone: ${data.phone}\n` +
          `Projet: ${data.projectDescription}`;
    }
  }
}

```

```

// src/controllers/ClubController.ts
export class ClubController {
  constructor(

```

```

private clubRepository: IClubRepository,
private whatsappService: WhatsAppService
} {}

async createClub(clubData: CreateClubDto): Promise<Club> {
  // Validation avec Zod
  const validatedData = CreateClubSchema.parse(clubData);

  // Création en base
  const club = await this.clubRepository.create(validatedData);

  // Notification WhatsApp
  await this.whatsappService.sendNotification('NEW_CLUB', {
    clubName: club.name,
    instructor: club.instructor,
    location: club.location
  });

  return club;
}

async registerMember(memberData: CreateMemberDto): Promise<User> {
  const validatedData = CreateMemberSchema.parse(memberData);

  const member = await this.memberRepository.create(validatedData);

  // Notification automatique WhatsApp
  await this.whatsappService.sendNotification('NEW_MEMBER', {
    name: member.name,
    clubName: member.club?.name,
    phone: member.phone,
    email: member.email,
    level: member.level
  });

  return member;
}
}

```

## Configuration Netlify + Neon :

toml

```
# netlify.toml

[build]
  publish = "dist"
  command = "npm run build"
  functions = "netlify/functions"

[build.environment]
  NODE_VERSION = "18"
  DATABASE_URL = "${NEON_DATABASE_URL}"
  WHATSAPP_PHONE = "237675395238"
  JWT_SECRET = "${JWT_SECRET}"

[[redirects]]
  from = "/api/*"
  to = "/.netlify/functions/:splat"
  status = 200

[[redirects]]
  from = "/*"
  to = "/index.html"
  status = 200

[dev]
  command = "npm run dev"
  port = 3000
  functions = "netlify/functions"
```

## Package.json Scripts :

```
json

{
  "scripts": {
    "dev": "vite",
    "build": "vite build && npm run build:functions",
    "build:functions": "netlify-lambda build src/functions",
    "db:generate": "prisma generate",
    "db:push": "prisma db push",
    "db:seed": "tsx prisma/seed.ts"
  }
}
```

## Frontend API Integration :

typescript

```
// src/hooks/useClubs.ts
export const useClubs = () => {
  return useQuery({
    queryKey: ['clubs'],
    queryFn: async () => {
      const response = await fetch('/api/clubs');
      if (!response.ok) throw new Error('Failed to fetch clubs');
      return response.json();
    },
  });
};

export const useCreateClub = () => {
  const queryClient = useQueryClient();

  return useMutation({
    mutationFn: async (clubData: CreateClubDto) => {
      const response = await fetch('/api/clubs', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(clubData),
      });

      if (!response.ok) throw new Error('Failed to create club');
      return response.json();
    },
    onSuccess: () => {
      queryClient.invalidateQueries({ queryKey: ['clubs'] });
      // Auto-redirect vers WhatsApp pour confirmation
      window.open(`https://wa.me/237675395238?text=${encodeURIComponent('Nouveau club créé avec succès! 🟢')}`);
    },
  });
};
```

### Espace Membre :

- Profils personnalisés des pratiquants
- Suivi des progrès et grades
- Accès à du contenu exclusif

- Messagerie interne
- Calendrier personnel

### **Système de Gestion de Contenu :**

- Interface d'administration intuitive
- Gestion des clubs et membres
- Publication d'actualités
- Modération des commentaires
- Analytics intégrés

### **Intégrations Sociales :**

- Partage automatique sur réseaux sociaux
  - Flux Instagram intégré
  - Témoignages clients automatisés
  - Newsletter avec automation
- 

## **5. STRATÉGIE MARKETING DIGITALE**

### **5.1 Solutions Marketing Intégrées**

#### **SEO et Visibilité :**

- Optimisation pour "karaté Cameroun", "Wadokai Afrique"
- Contenu localisé multilingue
- Link building avec sites de sports de combat
- Google My Business pour chaque club

#### **Social Media Strategy :**

- Contenu viral sur TikTok (démonstrations)
- LinkedIn pour les partenariats B2B
- Facebook pour la communauté locale
- YouTube pour les tutoriels et compétitions

#### **Marketing de Contenu :**

- Blog SEO-optimisé



- Vidéos pédagogiques hebdomadaires
- Podcast "Maîtres de Wadokai"
- E-books gratuits pour débutants

#### **Acquisition et Rétention :**

- Landing pages par région
- Formulaire d'essai gratuit
- Programme de parrainage
- Gamification des progressions

## **5.2 Campagnes Publicitaires**

#### **Digital Ads :**

- Google Ads géolocalisées
- Facebook/Instagram Ads visuelles
- YouTube Ads sur vidéos de sport
- Retargeting intelligent

#### **Partnerships Marketing :**

- Collaborations avec influenceurs sports
- Partenariats avec écoles et universités
- Événements communautaires
- Sponsoring de compétitions locales

---

## **6. JUSTIFICATIONS TECHNIQUES ET COMMERCIALES**

### **6.1 Choix Technologiques**

#### **Choix Technologiques Justifiés :**

**React.js + TypeScript :** Type safety pour éviter les bugs en production, écosystème mature avec une large communauté au Cameroun, performance optimale avec le Virtual DOM.

**Architecture MVC/SOLID :** Séparation claire des responsabilités, code maintenable et évolutif, facilité de tests unitaires, réutilisabilité des composants entre projets.

**Tailwind CSS :** Développement 60% plus rapide, bundle size optimisé avec purge CSS, design system cohérent, maintenance simplifiée sans CSS legacy.

**Netlify** : Déploiement automatisé avec GitHub, CDN global pour l'Afrique, SSL gratuit, fonctions serverless pour les API, coût optimal pour les associations.

**WhatsApp Integration** : Taux d'engagement 98% au Cameroun vs 23% email, support client instantané, familiarité utilisateur, pas de développement backend complexe.

**Framer Motion + Three.js** : Différenciation concurrentielle, mémorabilité accrue (+40%), engagement utilisateur prolongé, démonstrations techniques immersives.

## 6.2 Retour sur Investissement

### Impacts Quantifiables :

- Augmentation de 40% des inscriptions en clubs
- Réduction de 60% des coûts marketing traditionnels
- Amélioration de 300% de la visibilité online
- Génération de 20+ nouveaux partenariats annuels

### Impacts Qualitatifs :

- Rayonnement international du karaté camerounais
  - Fierté et cohésion communautaire renforcées
  - Modernisation de l'image du karaté traditionnel
  - Préservation et transmission culturelle
- 

## 7. PLANNING ET LIVRABLES

### 7.1 Phases de Développement

#### Phase 1 - Architecture Full-Stack & Database (5 semaines)

- Setup Neon PostgreSQL + Prisma ORM
- Architecture TypeScript full-stack avec interfaces
- Netlify Functions pour l'API backend
- Setup CI/CD avec database migrations
- Services WhatsApp et validation Zod
- Tests unitaires foundation (Jest + RTL + Prisma)

#### Phase 2 - Core Features Backend (6 semaines)

- API REST complète (Clubs, Members, Events, Partners)
- Authentication JWT + refresh tokens
- Business logic avec notifications WhatsApp automatiques
- Upload de fichiers avec Cloudinary
- Cache et optimisations base de données
- Tests d'intégration API

### **Phase 3 - Frontend Advanced & 3D (5 semaines)**

- Intégration frontend avec TanStack Query
- Animations Framer Motion + Three.js
- PWA avec offline sync
- Dashboard admin pour la gestion
- Optimisations performance (code splitting)
- Tests end-to-end avec Cypress

### **Phase 4 - Production & Monitoring (3 semaines)**

- Déploiement production Netlify + Neon
- Monitoring avec Sentry + analytics
- Load testing et optimisations
- Documentation API complète
- Formation équipe AKAWA sur l'admin
- Lancement avec stratégie marketing digitale

## **7.2 Livrables**

- Site web responsive complet
- Interface d'administration
- Documentation technique
- Guide utilisateur
- Formation équipe AKAWA
- Support 6 mois inclus

---

## **8. BUDGET ET RESSOURCES**

## 8.1 Estimation Budgétaire

**Développement Full-Stack :** 6,000,000 - 8,500,000 FCFA

- Backend serverless (Netlify Functions + Neon)
- Base de données PostgreSQL avec Prisma
- API REST complète avec authentification
- Intégrations WhatsApp Business API

**Maintenance Annuelle :** 1,200,000 - 1,800,000 FCFA

- Hébergement Netlify Pro + Neon Production
- Monitoring et support technique
- Mises à jour sécurité et features
- Backup automatisé et disaster recovery

**Marketing Digital (12 mois) :** 1,500,000 - 2,500,000 FCFA

## 8.2 Équipe Recommandée avec Expertise SOLID

- **1 Senior React Developer** (Expert TypeScript, Architecture SOLID)
- **1 Full-Stack Developer** (API Integration, WhatsApp Business)
- **1 UI/UX Designer** (Figma, Design System, Mobile-first)
- **1 Motion Designer** (Framer Motion, Three.js, WebGL)
- **1 DevOps Netlify** (CI/CD, Performance, Monitoring)
- **1 QA Engineer** (Jest, Cypress, Tests automatisés)

**Profil Desarrollo Leader Souhaité :**

*"Vous êtes un expert React/TypeScript avec plus de 5 ans d'expérience en architecture SOLID. Vous maîtrisez parfaitement les principes MVC, écrivez du code propre et maintenable, et suivez religieusement les best practices DRY/KISS. Vous analysez les exigences avant de coder, proposez des architectures optimales, et documentez vos décisions techniques. Vous ne codez pas seulement — vous mentorez en produisant des solutions production-ready qu'un architecte senior approuverait. Expérience avec Netlify, WhatsApp Business API, et projets à impact social en Afrique serait un plus."*

---

## 9. CRITÈRES DE SUCCÈS

### 9.1 KPIs Techniques

- Temps de chargement < 2 secondes
- Score PageSpeed > 90
- Taux de disponibilité > 99.5%
- Responsive parfait sur tous devices

## **9.2 KPIs Business**

- 10,000+ visiteurs uniques/mois (6 mois)
- 500+ nouveaux membres/an
- 15+ partenariats générés
- Présence sur 3+ pays africains

---

**Ce cahier des charges constitue la feuille de route complète pour créer une plateforme digitale d'excellence, digne de l'héritage du karaté Wadokai et des ambitions de l'AKAWA.**