

# YA FAQIH



## Documentation Technique Complète

Version	1.0.0
Date	30 November 2025
Plateforme	Web Application (PWA)
Stack	Next.js + PostgreSQL + Claude AI
Auteur	Équipe Ya Faqih

# Table des Matières

1. Vue d'ensemble du système
2. Architecture technique
3. Stack technologique
4. Base de données
5. Authentification et sécurité
6. APIs et endpoints
7. Fonctionnalités principales
8. Système de caching (PWA)
9. Intégrations externes
10. Déploiement et infrastructure
11. Performance et optimisation
12. Monitoring et logs
13. Plan de maintenance
14. Évolutions futures

# 1. Vue d'Ensemble du Système

## 1.1 Description

Ya Faqih est une application web progressive (PWA) d'assistance islamique intelligente basée sur l'IA. Elle fournit des réponses authentiques basées sur le Coran et la Sunnah, avec des fonctionnalités avancées de boussole Qibla et de notifications de prière.

## 1.2 Objectifs Principaux

- Fournir des réponses islamiques authentiques via l'IA Claude
- Offrir des outils pratiques (Qibla, horaires de prière)
- Garantir l'accessibilité hors ligne via PWA
- Assurer la sécurité et la confidentialité des données
- Optimiser les performances pour une expérience fluide

## 1.3 Utilisateurs Cibles

L'application s'adresse aux musulmans francophones et arabophones cherchant des réponses religieuses fiables, des outils de pratique quotidienne, et un accompagnement spirituel basé sur l'IA.

## 2. Architecture Technique

### 2.1 Architecture Globale

L'application suit une architecture client-serveur moderne avec les composants suivants : **Frontend (Client)** - Framework : Next.js 13+ (Pages Router) - UI : React 18 avec Tailwind CSS - État : React Hooks (useState, useEffect, useContext) - PWA : Service Workers pour le caching **Backend (Serveur)** - API Routes : Next.js API Routes - Runtime : Node.js 18+ - ORM : Prisma **Base de Données** - PostgreSQL (Neon Serverless) - Connexion pooling pour optimisation **Services Externes** - Anthropic Claude AI (API) - Al-Adhan API (horaires de prière) - NextAuth.js (authentification)

### 2.2 Flux de Données

Composant	Rôle	Technologies
Client Web	Interface utilisateur	React, Tailwind CSS
Service Worker	Cache et offline	Workbox, Cache API
API Routes	Logic métier	Next.js API, Node.js
Prisma ORM	Accès données	Prisma Client
PostgreSQL	Stockage persistant	Neon Database
Claude AI	Génération réponses	Anthropic API
Al-Adhan	Horaires prière	REST API

### 2.3 Patterns de Conception

- **MVC Pattern** : Séparation modèle-vue-contrôleur
- **Repository Pattern** : Abstraction accès données via Prisma
- **Service Pattern** : Logic métier dans services dédiés
- **Middleware Pattern** : Authentification et rate limiting
- **Cache-Aside Pattern** : Stratégies de caching PWA

## 3. Stack Technologique

### 3.1 Frontend

Technologie	Version	Usage
Next.js	14.0+	Framework React, SSR, API Routes
React	18.0+	Library UI composants
Tailwind CSS	3.3+	Framework CSS utility-first
Lucide React	0.263+	Icônes
NextAuth.js	4.24+	Authentification
Workbox	7.0+	Service Worker

### 3.2 Backend

Technologie	Version	Usage
Node.js	18.0+	Runtime JavaScript
Prisma	5.6+	ORM base de données
bcryptjs	2.4+	Hachage mots de passe
@anthropic-ai/sdk	Latest	Client API Claude

### 3.3 Base de Données

**PostgreSQL 15** hébergé sur Neon (serverless) avec les avantages suivants :

- Scalabilité automatique
- Connexion pooling intégré
- Backup automatique quotidien
- Latence optimisée
- Support des transactions ACID

## 4. Base de Données

### 4.1 Schéma de Données

#### Modèle User

```
model User { id String @id @default(cuid()) email String @unique name String? password String? emailVerified DateTime? image String? subscriptionTier String @default("free") messageCount Int @default(0) lastResetDate DateTime @default(now()) createdAt DateTime @default(now()) updatedAt DateTime @updatedAt accounts Account[] sessions Session[] conversations Conversation[] }
```

#### Modèle Conversation

```
model Conversation { id String @id @default(cuid()) title String userId String createdAt DateTime @default(now()) updatedAt DateTime @updatedAt user User @relation(fields: [userId], references: [id], onDelete: Cascade) messages Message[] @@index([userId]) }
```

#### Modèle Message

```
model Message { id String @id @default(cuid()) conversationId String role String content String @db.Text references String? @db.Text createdAt DateTime @default(now()) conversation Conversation @relation(fields: [conversationId], references: [id], onDelete: Cascade) @@index([conversationId]) }
```

### 4.2 Relations

Relation	Type	Description
User → Conversation	1:N	Un utilisateur a plusieurs conversations
Conversation → Message	1:N	Une conversation contient plusieurs messages
User → Account	1:N	OAuth providers multiples
User → Session	1:N	Sessions actives multiples

### 4.3 Index et Optimisations

Les index suivants sont créés pour optimiser les requêtes fréquentes :

- **User.email** : UNIQUE index pour authentification rapide
- **Conversation.userId** : Index pour récupération conversations
- **Message.conversationId** : Index pour chargement messages
- **Session.sessionToken** : Index pour validation session

## 5. Authentification et Sécurité

### 5.1 Système d'Authentification

L'application utilise **NextAuth.js** avec deux méthodes d'authentification :

Méthode	Provider	Description
OAuth 2.0	Google	Authentification via compte Google
Credentials	Email/Password	Authentification traditionnelle avec bcryptjs

### 5.2 Sécurité des Mots de Passe

• **Hachage** : bcryptjs avec salt rounds = 10 • **Validation** : Minimum 8 caractères • **Stockage** : Hash uniquement, jamais en clair • **Vérification** : bcrypt.compare() pour authentification

### 5.3 Sessions et Tokens

Les sessions sont gérées par NextAuth avec les caractéristiques suivantes :

- **JWT Strategy** : Tokens signés avec secret
- **Durée** : 30 jours d'inactivité
- **Refresh** : Automatique lors de l'utilisation
- **HttpOnly Cookies** : Protection XSS
- **Secure Flag** : HTTPS uniquement en production

### 5.4 Rate Limiting

Protection contre les abus avec **lru-cache** :

Endpoint	Limite	Fenêtre
/api/chat	10 requêtes	1 minute
/api/auth/signup	5 comptes	1 heure
/api/conversations/*	30 requêtes	1 minute

### 5.5 Protection CSRF et XSS

- **CSRF Tokens** : NextAuth intégré
- **Sanitization** : Échappement automatique React
- **Content Security Policy** : Headers sécurisés
- **HTTPS Only** : Redirection automatique
- **Validation Entrées** : Côté client et serveur

## 6. APIs et Endpoints

### 6.1 API Routes

Endpoint	Méthode	Description	Auth
/api/auth/[...nextauth]	GET/POST	NextAuth handlers	Public
/api/auth/signup	POST	Création compte	Public
/api/chat	POST	Envoyer message IA	Requis
/api/conversations/list	GET	Liste conversations	Requis
/api/conversations/[id]	GET	Récupérer conversation	Requis
/api/conversations/[id]	DELETE	Supprimer conversation	Requis
/api/cron/reset-messages	GET	Reset compteurs Pro	Cron

### 6.2 Format des Requêtes/Réponses

#### Exemple : POST /api/chat

```
Request: { "message": "■■ ■■ ■■■■■ ■■■■■■■■", "subscriptionTier": "free",  
  "userId": "clx123..." } Response: { "response": "■■■■■ ■■■■■ ■■■■■  
■■...", "references": [ "■■■■■ ■■■■■■■■: 78"], "conversationId":  
  "conv_abc123", "usage": { "messagesUsed": 5, "messagesLimit": 10, "tier":  
  "free" } }
```

### 6.3 Codes d'Erreur

Code	Signification	Action
400	Bad Request	Données invalides
401	Unauthorized	Session expirée
403	Forbidden	Limite atteinte
404	Not Found	Ressource inexistante
405	Method Not Allowed	Méthode incorrecte
429	Too Many Requests	Rate limit dépassé
500	Server Error	Erreur serveur



## 7. Fonctionnalités Principales

### 7.1 Chat Intelligent avec IA

**Claude Sonnet 4** fournit des réponses islamiques authentiques avec : • **Références** : Citations du Coran et Hadiths • **Badges d'authenticité** : ■■■■, ■■■■, ■■■■, ■■■■ • **Contextualisation** : Explications détaillées selon tier • **Multilingue** : Arabe principalement, français accepté • **Streaming** : Réponses progressives (futurs) **Limites par Tier** : • Free : 10 messages par session, 1000 tokens max • Pro : 100 messages par mois, 2000 tokens max • Premium : Illimité, 4000 tokens max

### 7.2 Boussole Qibla

**Fonctionnalités** : • Calcul précis direction La Mecque via géolocalisation • Boussole interactive avec orientation temps réel • Vibrations progressives (forte/moyenne/légère) • Animation Kaaba quand pointé correctement ( $\pm 15^\circ$ ) • Fonctionne hors ligne **Technologies** : • Geolocation API • DeviceOrientation API • Vibration API • Calcul géodésique bearing

### 7.3 Notifications de Prière

**5 Prières Quotidiennes** : ■■■■ (Fajr), ■■■■ (Dhuhr), ■■■■ (Asr), ■■■■ (Maghrib), ■■■■ (Isha) **Fonctionnement** : • Récupération horaires via Al-Adhan API • Notifications 5 minutes avant + au moment exact • Cache horaires 24 heures (offline) • Géolocalisation automatique • Badge clignotant si prochaine prière < 60 min **Méthode de Calcul** : ISNA (Islamic Society of North America)

### 7.4 Export PDF

**Premium uniquement** - Export conversations en PDF avec formatage professionnel, logo, date, et préservation des références.

### 7.5 Mode Sombre

Toggle dark/light mode avec persistance localStorage et détection préférence système.

### 7.6 Historique et Favoris

- Sauvegarde automatique conversations (1 par jour)
- Chargement conversations passées
- Système de favoris pour messages importants
- Suppression conversations
- Recherche dans l'historique (futur)

## 8. Système de Caching (PWA)

### 8.1 Service Worker

Le Service Worker (sw.js) implémente 4 stratégies de caching intelligentes :

Stratégie	Usage	Durée Cache
Cache First	Assets statiques (JS, CSS, images)	7 jours
Network First	API internes (/api/*)	5 minutes
Stale-While-Revalidate	API Al-Adhan (horaires prière)	24 heures
Cache + Refresh	Pages dynamiques	1 heure

### 8.2 Fonctionnalités Offline

- Page offline.html élégante en arabe
- Conversations sauvegardées accessibles
- Boussole Qibla fonctionnelle
- Horaires prière en cache (24h)
- Détection automatique perte connexion

### 8.3 Gestion des Mises à Jour

Le composant UpdateNotification informe l'utilisateur des nouvelles versions disponibles avec possibilité de mettre à jour immédiatement ou reporter.

### 8.4 Performance Gains

Métrique	Avant Cache	Après Cache	Gain
First Load	3.2s	1.2s	-62%
Time to Interactive	4.1s	1.5s	-63%
API Response	500ms	50ms	-90%
Repeat Visit	2.8s	0.3s	-89%

## 9. Intégrations Externes

### 9.1 Anthropic Claude AI

**API** : @anthropic-ai/sdk **Modèle** : claude-sonnet-4-20250514 **Endpoint** : https://api.anthropic.com/v1/messages **Configuration** : • Max tokens : 1000 (Free), 2000 (Pro), 4000 (Premium) • System prompt adapté par tier • Streaming désactivé (futur) • Retry logic avec exponential backoff **Coûts Estimés** : • Input : \$3 / 1M tokens • Output : \$15 / 1M tokens • Moyenne : \$0.01 par conversation

### 9.2 AI-Adhan API

**API** : https://aladhan.com/prayer-times-api **Endpoint** : GET /v1/timings/:timestamp **Paramètres** : • latitude, longitude : Géolocalisation utilisateur • method : 2 (ISNA) • Gratuit, pas de clé API **Réponse** : JSON avec horaires des 5 prières + Sunrise, Imsak, Midnight Cache 24 heures pour réduire requêtes

### 9.3 Google OAuth

**Provider** : NextAuth Google Provider **Scopes** : email, profile **Configuration** : • Client ID : Variable d'environnement • Client Secret : Variable d'environnement sécurisée • Callback URL : /api/auth/callback/google • Consent screen : Branding Ya Faqih

## 10. Déploiement et Infrastructure

### 10.1 Hébergement

Composant	Provider	Plan
Application (Frontend + API)	Vercel	Hobby/Pro
Base de Données	Neon	Free/Scale
Stockage Fichiers	Vercel Blob	Hobby
DNS	Vercel	Inclus

### 10.2 Variables d'Environnement

**Production (.env.production) :**

- **DATABASE\_URL :** Connection string PostgreSQL Neon
- **NEXTAUTH\_URL :** <https://www.yafaqih.app>
- **NEXTAUTH\_SECRET :** Secret JWT (généré)
- **ANTHROPIC\_API\_KEY :** Clé API Claude
- **GOOGLE\_CLIENT\_ID :** OAuth Google
- **GOOGLE\_CLIENT\_SECRET :** OAuth Google
- **CRON\_SECRET :** Token sécurisation cron

**Développement (.env.local) :** Mêmes variables avec valeurs de test

### 10.3 CI/CD Pipeline

**Git Push → Vercel Auto Deploy**

1. **Build :** next build
2. **Tests :** Lint, Type check
3. **Deploy Preview :** URL temporaire pour test
4. **Deploy Production :** Après merge dans main
5. **Cache Invalidation :** Automatique

**Durée moyenne :** 2-3 minutes **Rollback :** Instantané via Vercel dashboard

### 10.4 Domaine et SSL

- **Domaine :** [www.yafaqih.app](https://www.yafaqih.app)
- **SSL :** Let's Encrypt (automatique via Vercel)
- **Redirection :** HTTP → HTTPS
- **WWW :** Redirection [yafaqih.app](https://www.yafaqih.app) → [www.yafaqih.app](https://www.yafaqih.app)

# 11. Performance et Optimisation

## 11.1 Lighthouse Scores

Métrique	Score	Cible
Performance	95	90+
Accessibility	100	90+
Best Practices	95	90+
SEO	100	90+
PWA	90	90+

## 11.2 Optimisations Appliquées

- **Code Splitting** : Chargement lazy des composants
- **Image Optimization** : Next.js Image avec WebP
- **Font Optimization** : Preload fonts critiques
- **Tree Shaking** : Suppression code mort
- **Minification** : JS, CSS automatique
- **Compression Gzip** : Activée sur Vercel
- **CDN Edge** : Assets servis depuis edge network
- **Database Indexing** : Index sur colonnes fréquentes

## 11.3 Web Vitals

Métrique	Valeur	Cible	Status
LCP (Largest Contentful Paint)	1.2s	<2.5s	✓
FID (First Input Delay)	45ms	<100ms	✓
CLS (Cumulative Layout Shift)	0.05	<0.1	✓
TTFB (Time to First Byte)	180ms	<600ms	✓

## 12. Monitoring et Logs

### 12.1 Logging

**Console Logs** (Développement) : • Niveau : DEBUG • Format : Timestamp + Level + Message **Vercel Logs** (Production) : • Niveau : INFO, WARN, ERROR • Rétenion : 7 jours (Hobby), 30 jours (Pro) • Accès : Vercel Dashboard → Logs **Logs Critiques** : • Erreurs API Claude : Status + Message • Rate Limit Hit : IP + Endpoint • Auth Failures : Email (masqué) • Database Errors : Query (sans données sensibles)

### 12.2 Monitoring Recommandé

- **Vercel Analytics** : Web Vitals, Traffic
- **Sentry** (optionnel) : Error tracking
- **Uptime Robot** : Availability monitoring
- **Google Analytics** : User behavior

### 12.3 Alertes

Configurer alertes pour :

- Downtime > 5 minutes
- Error rate > 5%
- API response time > 2s
- Database connection failures

## 13. Plan de Maintenance

### 13.1 Maintenance Régulière

Tâche	Fréquence	Responsable
Mise à jour dépendances	Mensuel	Dev
Backup base de données	Quotidien	Neon Auto
Vérification logs erreurs	Hebdomadaire	Dev
Test fonctionnalités critiques	Hebdomadaire	QA
Revue performance	Mensuel	Dev
Mise à jour documentation	Trimestriel	Team

### 13.2 Procédure de Mise à Jour

1. **Development** : Branche feature/fix 2. **Testing** : Tests locaux + preview deployment 3. **Review** : Code review (si équipe) 4. **Merge** : Vers main après validation 5. **Deploy** : Automatique Vercel 6. **Smoke Test** : Vérification fonctionnalités critiques 7. **Monitoring** : Surveillance 24h post-deploy

### 13.3 Rollback Procedure

**En cas de problème critique** : 1. Identifier le déploiement problématique 2. Vercel Dashboard → Deployments 3. Sélectionner déploiement précédent stable 4. Cliquer "Promote to Production" 5. Confirmer rollback (instantané) 6. Investiguer cause en dev 7. Corriger et redéployer

## 14. Évolutions Futures

### 14.1 Court Terme (3 mois)

- **Icônes PWA complètes** : Score Lighthouse PWA 100/100
- **Son Adhan** : Audio pour notifications prière
- **Amélioration UI/UX** : Feedback utilisateurs
- **Tests E2E** : Cypress ou Playwright
- **Monitoring avancé** : Sentry integration

### 14.2 Moyen Terme (6 mois)

- **Calendrier Hijri** : Dates islamiques importantes
- **Compteur Tasbeih** : Digital dhikr counter
- **99 Noms d'Allah** : Avec explications
- **Lecture Coran** : Audio avec récitation
- **Mode famille** : Contrôle parental
- **Multilangue** : Anglais, Espagnol

### 14.3 Long Terme (12+ mois)

- **App Mobile Native** : React Native iOS/Android
- **Communauté** : Forums et partage
- **Enseignants** : Plateforme cours en ligne
- **Zakat Calculator** : Calcul automatique
- **Halal Scanner** : Vérification produits
- **Marketplace** : Produits islamiques

### 14.4 Roadmap Technique

Feature	Priorité	Effort	Impact
Icônes PWA	Haute	Faible	Moyen
Son Adhan	Haute	Faible	Élevé
Tests E2E	Moyenne	Moyen	Élevé
Calendrier Hijri	Moyenne	Moyen	Moyen
App Mobile	Basse	Élevé	Élevé
Multilangue	Moyenne	Élevé	Élevé



# Conclusion

**Ya Faqih** représente une solution moderne et complète pour l'assistance islamique intelligente. L'architecture robuste, les mesures de sécurité avancées, et les optimisations de performance garantissent une expérience utilisateur exceptionnelle. **Points Forts :**

- Architecture scalable et maintenable
- Sécurité multi-niveaux (auth, rate limiting, HTTPS)
- Performance optimisée (Lighthouse 95+)
- PWA avec fonctionnalités offline
- Intégrations API fiables (Claude, Al-Adhan)
- Base de données optimisée avec indexation
- Déploiement automatisé via CI/CD

**Prochaines Étapes Recommandées :**

1. Compléter les icônes PWA pour score 100/100
2. Implémenter monitoring avancé (Sentry)
3. Ajouter tests E2E pour régression
4. Intégrer son Adhan pour notifications
5. Planifier évolutions calendrier Hijri

Cette documentation doit être mise à jour régulièrement pour refléter les évolutions de l'application et servir de référence pour l'équipe de développement et maintenance.