YAKO AFRICA Assurances Vie

YNOV – Plateforme Front-Office

DOCUMENTATION COMPLÈTE

SYSTÈME D'AUTHENTIFICATION

★ **VERSION 2.0** ★

15 Juin 2026

Chef de projet : Alexia AKE | Lead Backend : Bruce YAPO

Rôle	Nom	Email
Chef de projet	Alexia AKE	alexia.ake@yakoafricassur.com
Lead Backend	Bruce YAPO	bruce.yapo@yakoafricassur.com
Lead Frontend	À définir	-

TABLE DES MATIÈRES

1. PRÉSENTATION GÉNÉRALE

1.1 Objectif du Projet

Le système d'authentification YNOV est une solution complète de gestion des accès pour la plateforme front-office de YAKO AFRICA Assurances Vie. Il permet :

- La gestion sécurisée des comptes utilisateurs (clients, agents, administrateurs)
- L'authentification multi-rôles avec permissions granulaires
- Le suivi des sessions et de l'activité des utilisateurs
- La protection contre les attaques par force brute avec gel progressif à 3 niveaux
- La gestion du cycle de vie des mots de passe
- Le dégel automatique intelligent sans intervention admin

1.2 Technologies Utilisées

Composant	Technologie	Version
Backend	Laravel	10.x
Frontend	HTML5/CSS3/JavaScript	ES6
Base de données	MySQL	8.0+
Authentification API	Laravel Sanctum	-
Gestion des rôles	Spatie Permission	5.x
Styles CSS	Personnalisé	-
Requêtes HTTP	Axios	-
Notifications	SweetAlert2	-

1.3 URLs de Production

★ NOUVEAUTÉ v2.0 — Environnement de déploiement actif

Environnement	URL
API Backend	https://apidev.yakoafricassur.com
Documentation API interactive	https://apidev.yakoafricassur.com/docs (accueil de l'api)
Frontend Test	https://api-auth-ynov.yakoafricassur.com

1.4 Couleurs de la Charte Graphique

Variable CSS	Code Hex	Usage
--brand-primary	#1D603D	Vert YAKO – couleur principale
--brand-primary-dark	#0B482F	Vert foncé
--brand-accent	#E09518	Orange YAKO – couleur d'accent

--brand-success	#1D603D	Succès
--brand-warning	#E09518	Avertissement
--brand-danger	#DC3545	Erreur / Danger

2. ARCHITECTURE DU SYSTÈME

2.1 Architecture Globale

Le système est organisé en trois couches distinctes :

- Frontend (HTML/CSS/JS) : https://api-auth-ynov.yakoafricassur.com
- Backend (Laravel 10) : https://apidev.yakoafricassur.com/api
- Base de données (MySQL) : persistance de toutes les données

Couche	Composants
Middlewares	CORS, RateLimit, CheckAccount, CheckInactivity
Controllers	AuthController, PasswordController, SessionController, CronController, DashboardController, UserController
Services	AuthService, PasswordService, MailService, UserStatusService
Models	User, Profile, SessionsLog, LoginAttempt, ScheduledUnfreeze, PasswordHistory, UserStatusHistory

2.2 Flux des Données

Flux	Description
1. Inscription	User → register.html → API POST /auth/register → Email de vérification
2. Vérification Email	User → email link → API GET /auth/verify-email → Compte activé
3. Connexion	User → login.html → API POST /auth/login → Token JWT → Dashboard
4. Gel Compte ★ NOUVEAU	2 tentatives → gel 10s / 3 tentatives → gel 3min / 5 tentatives → gel 30min
5. Dégel Auto ★ NOUVEAU	Cron JS → API GET /cron/check-unfreeze → Dégel automatique
6. Gestion Session	Frontend → API POST /session/extend → Token prolongé
7. Déconnexion	User → API POST /auth/logout → Token révoqué → Redirection login

3. BASE DE DONNÉES

3.1 Table : users

Champ	Type	Description	Contraintes
id	BIGINT	Identifiant technique	AUTO_INCREMENT, PK
uuid	CHAR(36)	Identifiant unique	UNIQUE
login	VARCHAR(255)	Login utilisateur	UNIQUE
email	VARCHAR(255)	Email	UNIQUE
password	VARCHAR(255)	Mot de passe hashé	-
status	ENUM	Statut du compte	actif, inactif, gele, bloque
user_type	ENUM	Type d'utilisateur	client, user_interne, user_partner, admin, autre
is_first_login	BOOLEAN	Première connexion	DEFAULT true
password_changed_at	TIMESTAMP	Date dernier changement	NULL
last_login_at	TIMESTAMP	Dernière connexion	NULL
last_activity_at	TIMESTAMP	Dernière activité	NULL
is_online	BOOLEAN	En ligne	DEFAULT false
email_verified_at	TIMESTAMP	Email vérifié	NULL
created_at	TIMESTAMP	Date création	-
updated_at	TIMESTAMP	Date mise à jour	-
deleted_at	TIMESTAMP	Soft delete	NULL

3.2 Table : profiles

Champ	Type	Description
id	BIGINT	Identifiant technique (PK)
uuid	CHAR(36)	Identifiant unique
user_uuid	CHAR(36)	FK vers users.uuid
code_agent	VARCHAR(50)	Code agent unique
nom	VARCHAR(100)	Nom de famille
prenoms	VARCHAR(100)	Prénoms
mobile	VARCHAR(20)	Téléphone mobile
telephone	VARCHAR(20)	Téléphone fixe
email_secondaire	VARCHAR(255)	Email secondaire
fonction	VARCHAR(100)	Fonction
agence_uuid	VARCHAR(36)	UUID agence
agence_libelle	VARCHAR(200)	Libellé agence
partner_uuid	VARCHAR(36)	UUID partenaire

branche	VARCHAR(100)	Branche
photo	VARCHAR(255)	Chemin photo
date_naissance	DATE	Date naissance
lieu_naissance	VARCHAR(100)	Lieu naissance

3.3 Table : sessions_logs

Champ	Type	Description
id	BIGINT	Identifiant technique (PK)
user_uuid	CHAR(36)	FK vers users.uuid
session_id	VARCHAR(255)	ID session unique
ip_address	VARCHAR(45)	Adresse IP
user_agent	TEXT	User agent navigateur
login_at	TIMESTAMP	Date connexion
last_activity_at	TIMESTAMP	Dernière activité
logout_at	TIMESTAMP	Date déconnexion

3.4 Table : login_attempts

Champ	Type	Description
id	BIGINT	Identifiant technique (PK)
identifier	VARCHAR(255)	Email ou login tenté
ip_address	VARCHAR(45)	Adresse IP source
was_successful	BOOLEAN	Tentative réussie
attempted_at	TIMESTAMP	Date tentative
attempt_count	INT	Nombre de tentatives

3.5 Table : scheduled_unfreezes

★ NOUVEAUTÉ v2.0 — Nouvelle table pour le dégel automatique à 3 niveaux

Champ	Type	Description
id	BIGINT	Identifiant technique (PK)
user_uuid	CHAR(36)	FK vers users.uuid
unfreeze_at	TIMESTAMP	Date et heure de dégel prévu
duration_seconds	INT	Durée du gel en secondes
freeze_level	INT	Niveau de gel (1, 2 ou 3)
notification_sent	BOOLEAN	Notification email envoyée
created_at	TIMESTAMP	Date création

updated_at	TIMESTAMP	Date mise à jour

3.6 Table : password_histories

★ NOUVEAUTÉ v2.0 — Nouvelle table pour l'historique des mots de passe (réutilisation interdite)

Champ	Type	Description
id	BIGINT	Identifiant technique (PK)
user_uuid	CHAR(36)	FK vers users.uuid
password_hash	VARCHAR(255)	Hash de l'ancien mot de passe
created_at	TIMESTAMP	Date création

3.7 Table : user_status_histories

Champ	Type	Description
id	BIGINT	Identifiant technique (PK)
user_uuid	CHAR(36)	FK vers users.uuid
changed_by	VARCHAR(36)	UUID de l'acteur ou 'system'
old_status	VARCHAR(20)	Ancien statut
new_status	VARCHAR(20)	Nouveau statut
reason	TEXT	Motif du changement
ip_address	VARCHAR(45)	Adresse IP source
created_at	TIMESTAMP	Date changement

4. BACKEND LARAVEL

4.1 Structure des Dossiers

```
app/
├── Console/Commands/
│   ├── ProcessAutoUnfreezes.php
│   └── CheckPasswordExpiry.php
├── Http/
│   ├── Controllers/
│   │   ├── Auth/
│   │   │   ├── AuthController.php
│   │   │   ├── PasswordController.php
│   │   │   └── SessionController.php
│   │   ├── Admin/
│   │   │   └── UserController.php
│   │   ├── CronController.php          ★ NOUVEAU
│   │   └── DashboardController.php
│   └── Middleware/
│       ├── CheckAccountStatus.php
│       ├── CheckInactivity.php
│       └── RateLimitMiddleware.php
├── Models/
│   ├── User.php
│   ├── Profile.php
│   ├── SessionsLog.php
│   ├── LoginAttempt.php
│   ├── ScheduledUnfreeze.php           ★ NOUVEAU
│   └── UserStatusHistory.php
└── Services/
    ├── AuthService.php
    ├── UserStatusService.php
    ├── PasswordService.php
    └── MailService.php
```

4.2 Controllers Principaux

AuthController

Méthode	Route	Description
register()	POST /api/auth/register	Inscription utilisateur
login()	POST /api/auth/login	Connexion
logout()	POST /api/auth/logout	Déconnexion
verifyEmail()	GET /api/auth/verify-email	Vérification email
resendVerification()	POST /api/auth/resend-verification	Renvoi email vérification

PasswordController

Méthode	Route	Description
forgot()	POST /api/auth/forgot-password	Demande reset
reset()	POST /api/auth/reset-password	Reset mot de passe

change()	POST /api/auth/change-password	Changement (connecté)
checkExpiry()	GET /api/auth/check-password-expiry	Vérification expiration

SessionController

Méthode	Route	Description
current()	GET /api/session/current	Session actuelle
extend()	POST /api/session/extend	Prolonger session
activeSessions()	GET /api/session/active	Sessions actives
terminate()	DELETE /api/session/terminate/{id}	Terminer session
checkStatus()	GET /api/session/check-status	État session

CronController — NOUVEAU

★ NOUVEAUTÉ v2.0 — Endpoints publics de gestion du dégel automatique

Méthode	Route	Description
checkAndUnfreeze()	GET /api/cron/check-unfreeze	Vérifier et dégeler compte (param: login)
autoUnfreezeSelf()	POST /api/cron/auto-unfreeze	Forcer dégel (param: login)
getFreezeStatus()	GET /api/cron/freeze-status/{uuid}	Statut de gel d'un utilisateur
getFrozenAccounts()	GET /api/cron/frozen-accounts	Liste de tous les comptes gelés
checkUnfreezes()	GET /api/cron/check-unfreezes	Traiter les dégels en attente

4.3 Services

UserStatusService – Gel Progressif à 3 Niveaux

★ NOUVEAUTÉ v2.0 — Nouvelle logique de gel progressif avec 3 seuils distincts

Élément	Valeur / Description
STATUS_ACTIF	actif
STATUS_INACTIF	inactif
STATUS_GEL	gele
STATUS_BLOQUE	bloque
FREEZE_LEVEL_1_THRESHOLD	2 tentatives → gel 10 secondes
FREEZE_LEVEL_2_THRESHOLD	3 tentatives → gel 3 minutes
FREEZE_LEVEL_3_THRESHOLD	5 tentatives → gel 30 minutes

FREEZE_LEVEL_1_DURATION	10 secondes
FREEZE_LEVEL_2_DURATION	180 secondes (3 min)
FREEZE_LEVEL_3_DURATION	1800 secondes (30 min)
handleFailedAttempt()	Traite tentative échouée et applique gel progressif
autoUnfreeze()	Dégel auto : UPDATE niveaux 1-2, DELETE niveau 3
freeze()	Gèle un compte avec durée spécifique
unfreeze()	Dégel manuel (admin)
checkLoginAccess()	Vérifie si le compte peut se connecter

4.4 Middlewares

Middleware	Description	Routes protégées
CheckAccountStatus	Vérifie si le compte est actif	Toutes routes auth
CheckInactivity	Expire la session après 15 min	Toutes routes auth
RateLimitMiddleware	Limite tentatives (3/30 min)	POST /login

Note : CheckFirstLogin et CheckPasswordExpiry ont été intégrés directement dans AuthController et PasswordController.

5. FRONTEND HTML/CSS/JS (Juste pour tester l'API)

5.1 Structure des Pages

Fichier	Description
index.html	Page d'accueil / présentation
login.html	Page de connexion
register.html	Page d'inscription
forgot-password.html	Mot de passe oublié
reset-password.html	Réinitialisation du mot de passe
verify-email.html	Confirmation d'email
change-password.html	Changement de mot de passe (connecté)
dashboard.html	Tableau de bord principal
admin-dashboard.html ★ NOUVEAU	Dashboard administrateur
assets/css/ynov.css	Feuille de styles principale
assets/js/ynov-api.js	Client API (axios)
assets/js/ynov-auth.js	Gestionnaire d'authentification
assets/js/ynov-cron.js ★ NOUVEAU	Gestionnaire de dégel automatique
assets/js/password-expiry-checker.js ★ NOUVEAU	Vérificateur expiration MDP

5.2 Classes CSS Principales

Classe	Usage
.auth-container	Conteneur principal d'authentification
.auth-brand-panel	Panneau latéral gauche (marque)
.auth-form-panel	Panneau formulaire droit
.auth-card	Carte formulaire
.form-group	Groupe de champ
.form-control	Champ input/select
.form-label	Label de champ
.input-icon-wrapper	Wrapper avec icône
.invalid-feedback	Message d'erreur inline
.btn / .btn-primary	Bouton de base / Bouton vert YAKO
.btn-accent / .btn-outline	Bouton orange YAKO / Bouton contour
.btn-block	Bouton pleine largeur
.alert / .alert-success / .alert-error	Alerte et variantes
.topbar / .sidebar	Barre supérieure / latérale dashboard
.stat-card / .badge	Carte statistique / Badge de statut

5.3 API Client (ynov-api.js) – Méthodes Principales

Méthode	Description
register(data)	Inscription utilisateur
login(login, password)	Connexion
logout()	Déconnexion
forgotPassword(email)	Demande de réinitialisation
resetPassword(token, email, password, confirmation)	Réinitialisation du mot de passe
changePassword(currentPassword, newPassword)	Changement de mot de passe
checkPasswordExpiry()	Vérification expiration
getCurrentSession()	Session actuelle
extendSession()	Prolonger la session
getActiveSessions()	Sessions actives
terminateSession(sessionId)	Terminer une session
getDashboard()	Données du dashboard
getProfile() / updateProfile(data)	Profil utilisateur / Mise à jour
getNotifications()	Notifications
checkAndUnfreezeSelf(login) ★ NOUVEAU	Vérifier et dégeler le compte via cron
getFreezeStatus(userUuid) ★ NOUVEAU	Statut de gel d'un utilisateur
getFrozenAccounts() ★ NOUVEAU	Liste des comptes gelés (admin)
getUsers(params) / activateUser()	Liste et gestion utilisateurs (admin)
freezeUser() / unfreezeUser()	Geler / dégeler un compte (admin)
getAdminStats()	Statistiques admin

5.4 Cron JS (ynov-cron.js) — NOUVEAU

★ NOUVEAUTÉ v2.0 — Module de dégel automatique côté front-end

Le cron front-end vérifie périodiquement l'état de gel du compte utilisateur et déclenche le dégel automatique.

Fonctionnalité	Description
Intervalle adaptatif	Toutes les 10s si temps restant < 60s, toutes les 60s sinon
Dégel automatique	Appel API cron sans intervention admin
Affichage timer	Niveaux visuels distincts (level1, level2, level3)
Arrêt automatique	Le cron s'arrête dès que le compte est dégelé

```
class YnovCron {
  // Vérifie toutes les 10s (si temps < 60s) ou 60s (si temps >= 60s)
  // Dégèle automatiquement le compte quand le délai est écoulé
  // Met à jour l'affichage du timer avec niveaux visuels
}
```

6. API ENDPOINTS

6.1 Endpoints Publics

Méthode	Endpoint	Description	Paramètres
POST	/api/auth/register	Inscription	nom, prenoms, email, password, typ_user, role
POST	/api/auth/login	Connexion	login, password
POST	/api/auth/forgot-password	Mot de passe oublié	email
POST	/api/auth/reset-password	Réinitialisation	token, email, password, password_confirmation
GET	/api/auth/verify-email	Vérification email	token, email
POST	/api/auth/resend-verification	Renvoi email	email

6.2 Endpoints CRON — NOUVEAUX

★ NOUVEAUTÉ v2.0 — Routes publiques pour la gestion du dégel automatique

Méthode	Endpoint	Description
GET	/api/cron/check-unfreeze	Vérifier et dégeler compte (param: login)
POST	/api/cron/auto-unfreeze	Forcer dégel (param: login)
GET	/api/cron/freeze-status/{uuid}	Statut de gel d'un utilisateur
GET	/api/cron/frozen-accounts	Liste de tous les comptes gelés
GET	/api/cron/check-unfreezes	Traiter les dégels en attente

6.3 Endpoints Protégés (auth:sanctum)

Méthode	Endpoint	Description	Accès
POST	/api/auth/logout	Déconnexion	Tous
POST	/api/auth/change-password	Changer mot de passe	Tous
GET	/api/auth/check-password-expiry	Vérifier expiration	Agents

6.4 Endpoints Sessions

Méthode	Endpoint	Description
GET	/api/session/current	Session actuelle
POST	/api/session/extend	Prolonger la session
GET	/api/session/active	Sessions actives
DELETE	/api/session/terminate/{id}	Terminer une session
GET	/api/session/check-status	État de la session

6.5 Endpoints Dashboard

Méthode	Endpoint	Description
GET	/api/dashboard	Dashboard principal
GET	/api/dashboard/profile	Profil utilisateur
PUT	/api/dashboard/profile	Mettre à jour profil
GET	/api/dashboard/notifications	Notifications
GET	/api/dashboard/recent-activity	Activité récente

6.6 Endpoints Administration

Méthode	Endpoint	Description	Rôle
GET	/api/admin/users	Liste utilisateurs	admin / super_admin
GET	/api/admin/users/stats	Statistiques	admin / super_admin
GET	/api/admin/users/{uuid}	Détails utilisateur	admin / super_admin
POST	/api/admin/users/{uuid}/activate	Activer compte	admin / super_admin
POST	/api/admin/users/{uuid}/deactivate	Désactiver compte	admin / super_admin
POST	/api/admin/users/{uuid}/freeze	Geler compte	admin / super_admin
POST	/api/admin/users/{uuid}/unfreeze	Dégeler compte	admin / super_admin
GET	/api/admin/users/{uuid}/history	Historique statut	admin / super_admin

7. FLUX D'AUTHENTIFICATION

7.1 Flux d'Inscription

Étape	Acteur	Action	Résultat
1	Client	Soumet le formulaire register.html	Données envoyées à POST /api/auth/register
2	API	Validation des données	Erreur 422 si invalide
3	API	Création de l'utilisateur en base	Compte créé avec statut inactif
4	API	Assignation du rôle	Rôle Spatie associé
5	API	Création du profil	Enregistrement dans la table profiles
6	Email	Envoi de l'email de vérification	Token unique généré
7	Client	Clic sur le lien dans l'email	Requête GET /api/auth/verify-email
8	API	Vérification du token	email_verified_at mis à jour
9	Client	Redirection vers login.html	Compte actif, prêt à se connecter

7.2 Flux de Connexion

Étape	Acteur	Action	Résultat
1	Client	Soumet login + password via login.html	Requête POST /api/auth/login
2	API	Vérification des tentatives IP	Erreur 429 si trop de tentatives
3	API	Recherche de l'utilisateur en base	Erreur 401 si non trouvé
4	API	Vérification du mot de passe (bcrypt)	Erreur 401 si incorrect
5	API	Vérification du statut du compte	Erreur 403/423 si inactif/gelé/bloqué
6	API	Mise à jour last_login_at, is_online	Données utilisateur mises à jour
7	API	Création du log de session	Enregistrement dans sessions_logs
8	API	Génération du token Sanctum (8h)	Token retourné dans la réponse
9	Client	Stockage du token en localStorage	Token disponible pour les requêtes
10	Client	Redirection vers le dashboard	Accès accordé

7.3 Flux de Blocage Progressif à 3 Niveaux — NOUVEAU

★ NOUVEAUTÉ v2.0 — Gel progressif : 2 tentatives → 10s | 3 tentatives → 3min | 5 tentatives → 30min

Tentative	Compteur	Statut	Action système	Message utilisateur

1	1	Actif	Log de la tentative	Login ou mot de passe incorrect
2	2	GEL (10s)	Gel niveau 1 + planification	Compte gelé pour 10 secondes
3 (pendant gel)	2	Gelé	Avertissement pendant gel	Login incorrect. Veuillez patienter
3 (après gel)	2	GEL (3min)	Gel niveau 2 + planification	Compte gelé pour 3 minutes
4	2	Gelé	Avertissement pendant gel	Login incorrect. Veuillez patienter
5	3	GEL (30min)	Gel niveau 3 + Email envoyé	Compte gelé pour 30 minutes

7.4 Flux de Dégel Automatique — NOUVEAU

★ NOUVEAUTÉ v2.0 — Stratégie UPDATE/DELETE selon le niveau de gel

Niveau	Durée	Stratégie dégel	Historique tentatives	Planification BD
Niveau 1	10s	Dégel → attente niveau 2	CONSERVÉ	UPDATE
Niveau 2	3min	Dégel → attente niveau 3	CONSERVÉ	UPDATE
Niveau 3	30min	Dégel complet + Email	RÉINITIALISÉ	DELETE
Connexion réussie	-	Dégel immédiat	RÉINITIALISÉ	DELETE
Dégel manuel admin	-	Dégel immédiat	RÉINITIALISÉ	DELETE

8. GESTION DES UTILISATEURS

8.1 Statuts des Comptes

Statut	Code	Description	Connexion	Action requise
Actif	actif	Compte normal, fonctionnel	Autorisée	Aucune
Inactif	inactif	Compte désactivé manuellement	Refusée	Admin doit réactiver
Gelé	gele	Gel temporaire (10s/3min/30min)	Refusée	Attendre dégel auto ou admin
Bloqué	bloque	Blocage définitif	Refusée	Super Admin uniquement

8.2 Rôles et Permissions

Rôle	Permissions accordées
super_admin	Toutes les permissions du système
admin	view_users, create_users, edit_users, manage_users_status, manage_roles, view_clients, edit_clients, view_contracts, create_contracts, edit_contracts, view_claims, process_claims, view_dashboard
agent	view_clients, edit_clients, view_contracts, create_contracts, edit_contracts, view_claims, view_dashboard
client	view_contracts, view_dashboard

8.3 Types d'Utilisateurs

Type	Code	Description
Client	client	Client final de YAKO AFRICA
Utilisateur interne	user_interne	Employé YAKO AFRICA
Utilisateur partenaire	user_partner	Partenaire externe
Administrateur	admin	Administrateur de la plateforme
Autre	autre	Autres types d'utilisateurs

8.4 Actions Admin sur les Comptes

Action	Endpoint	Statuts concernés	Corps de la requête
Activation	POST .../activate	inactif/gele/bloque → actif	{ reason (optionnel) }
Désactivation	POST .../deactivate	actif → inactif	{ reason: string (requis) }
Gel temporaire	POST .../freeze	actif → gele	{ reason: string, duration_minutes: int }

Dégel	POST .../unfreeze	gele → actif	{ reason (optionnel) }

9. SÉCURITÉ

9.1 Mesures de Sécurité Implémentées

Mesure	Description	Paramètres
Rate Limiting	Limite les tentatives de connexion par IP	3 tentatives / 30 min
Blocage progressif	Gel 3 niveaux automatique	2 → 10s, 3 → 3min, 5 → 30min
Expiration mot de passe	Force le renouvellement périodique	90 jours (agents uniquement)
Première connexion	Force le changement du mot de passe initial	Obligatoire au premier login
Inactivité session	Expiration automatique en cas d'inactivité	15 minutes
Hash des mots de passe	Hachage sécurisé avec bcrypt	Par défaut Laravel
Tokens API	Authentification stateless via Sanctum	Expiration 8h
CORS	Restriction des origines cross-domain	Domaines autorisés uniquement
Validation email	Confirmation de l'adresse email requise	Token unique à usage unique
Historique mots de passe	Empêche la réutilisation	5 derniers mots de passe

9.2 Règles de Validation des Mots de Passe

Règle	Description
Longueur minimale	Minimum 8 caractères
Majuscule obligatoire	Au moins une lettre majuscule (A-Z)
Minuscule obligatoire	Au moins une lettre minuscule (a-z)
Chiffre obligatoire	Au moins un chiffre (0-9)
Caractère spécial	Au moins un caractère parmi : !@#$%^&*
Pas de répétitions	Aucun caractère répété de façon excessive
Pas de mots communs	Vérification anti-dictionnaire
Pas de séquences	Interdit : 12345678, azerty, etc.

9.3 Gestion des Tokens

Action	Méthode Laravel	Description
Création (8h)	$user->createToken(..., now()->addHours(8))	Génère un token avec expiration
Révocation courante	$user->currentAccessToken()->delete()	Supprime le token de session actif
Révocation totale	$user->tokens()->delete()	Supprime tous les tokens de l'utilisateur

9.4 En-têtes de Sécurité Recommandés

En-tête	Valeur	Rôle
X-Content-Type-Options	nosniff	Empêche le MIME sniffing
X-Frame-Options	DENY	Empêche le clickjacking
X-XSS-Protection	1; mode=block	Protection XSS basique
Strict-Transport-Security	max-age=31536000	Force HTTPS

10. INSTALLATION ET DÉPLOIEMENT

10.1 URLs de Production

Environnement	URL
API Backend	https://apidev.yakoafricassur.com
Documentation API interactive	https://apidev.yakoafricassur.com/docs (accueil de l'api)
Frontend Test	https://api-auth-ynov.yakoafricassur.com

10.2 Prérequis

Logiciel	Version minimale
PHP	>= 8.1
Composer	>= 2.0
MySQL	>= 8.0

10.3 Installation du Backend

```
# 1. Cloner le dépôt
git clone lien_depot
cd ynov-auth

# 2. Installer les dépendances
composer install

# 3. Copier et configurer .env
cp .env.example .env
php artisan key:generate

# 4. Configurer la base de données dans .env
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=ynov_auth
DB_USERNAME=root
DB_PASSWORD=

# 5. Exécuter les migrations et seeders
php artisan migrate
php artisan db:seed --class=RolePermissionSeeder

# 6. Créer le lien storage et démarrer
php artisan storage:link
php artisan serve --host=0.0.0.0 --port=8000
```

10.4 Configuration CORS – Production

```
APP_URL=https://apidev.yakoafricassur.com
```

```
FRONTEND_URL=https://api-auth-ynov.yakoafricassur.com
SANCTUM_STATEFUL_DOMAINS=api-auth-ynov.yakoafricassur.com
SESSION_DOMAIN=.yakoafricassur.com
```

10.5 Configuration Mail

```
MAIL_MAILER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=votre_username
MAIL_PASSWORD=votre_password
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS=noreply@ynov.com
MAIL_FROM_NAME=YNOV
```

10.6 Installation du Frontend

```
# 1. Modifier l'URL de l'API dans ynov-api.js
const API_BASE_URL = 'https://apidev.yakoafricassur.com/api';

# 2. Configuration Nginx (exemple production)
server {
    listen 443 ssl;
    server_name api-auth-ynov.yakoafricassur.com;
    root /var/www/ynov-frontend;
    index index.html;
    location / {
        try_files $uri $uri/ /index.html;
    }
}
```

11. DÉPANNAGE

11.1 Erreurs Courantes

Erreur	Cause probable	Solution
Session store not set on request	Utilisation de session() dans l'API	Remplacer par UUID
CORS Failed	Configuration CORS incorrecte	Modifier config/cors.php
Class not found	Autoloader non régénéré	composer dump-autoload
Connection refused	Serveur non démarré	php artisan serve
Invalid default value for 'timestamp'	Mode strict MySQL	Mettre 'strict' => false
Foreign key constraint fails	Ordre des migrations incorrect	Vérifier l'ordre de migration

11.2 Commandes de Nettoyage

```
php artisan config:clear
php artisan cache:clear
php artisan view:clear
php artisan route:clear
php artisan optimize:clear
chmod -R 775 storage bootstrap/cache
php artisan migrate:fresh --seed
```

11.3 Debug API

```
# Test connexion avec curl (production)
curl -X POST https://apidev.yakoafricassur.com/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{"login":"admin@yakoafricassur.com","password":"Admin@1234"}'

# Test cron dégel
curl https://apidev.yakoafricassur.com/api/cron/check-unfreeze?login=xxx

# Voir les logs Laravel
tail -f storage/logs/laravel.log
```

12. ANNEXES

12.1 Codes d'Erreur API

HTTP	Code API	Description
200	SUCCESS	Succès
201	CREATED	Créé avec succès
400	INVALID_REQUEST	Requête invalide
401	INVALID_CREDENTIALS	Identifiants invalides
401	INACTIVITY_TIMEOUT	Session expirée par inactivité
403	ACCOUNT_DEACTIVATED	Compte désactivé
403	ACCOUNT_BLOCKED	Compte bloqué définitivement
403	FIRST_LOGIN_REQUIRED	Première connexion obligatoire
403	PASSWORD_EXPIRED	Mot de passe expiré
423	ACCOUNT_FROZEN ★ NOUVEAU	Compte gelé temporairement (code HTTP 423)
404	NOT_FOUND	Ressource non trouvée
422	VALIDATION_ERROR	Erreur de validation
429	RATE_LIMIT_EXCEEDED	Trop de tentatives de connexion

12.2 Exemples de Réponses API

Login Succès (200)

```
{
  "success": true,
  "message": "Connexion réussie",
  "data": {
    "access_token": "1|xxxxxxxxxxxxxxxxxxx",
    "token_type": "Bearer",
    "expires_in": 28800,
    "user": {
      "uuid": "550e8400-e29b-41d4-a716-446655440000",
      "email": "agent@yakoafricassur.com",
      "role": "agent",
      "status": "actif",
      "is_first_login": false
    },
    "requires_password_change": false,
    "redirect_to": "/agent/dashboard"
  }
}
```

Compte Gelé (423) — NOUVEAU

```
{
  "success": false,
  "message": "Compte gelé temporairement pour 10 secondes.",
  "code": "ACCOUNT_FROZEN",
```

```
  "level": 1,
  "remaining_seconds": 10,
  "duration_readable": "10 secondes"
}
```

Check Unfreeze (GET /cron/check-unfreeze) — NOUVEAU

```
{
  "success": true,
  "is_frozen": true,
  "freeze_level": 1,
  "remaining_seconds": 5,
  "remaining_readable": "5 secondes",
  "unfreeze_at": "2026-06-15T15:30:05+00:00"
}
```

Rate Limit Dépassé (429)

```
{
  "success": false,
  "message": "Trop de tentatives. Réessayez dans 30 minutes.",
  "code": "RATE_LIMIT_EXCEEDED"
}
```

13. NOUVEAUTÉS VERSION 2.0

13.1 Gel Progressif à 3 Niveaux

Niveau	Seuil tentatives	Durée gel	Notification email
Niveau 1	2 tentatives	10 secondes	Non
Niveau 2	3 tentatives	3 minutes	Non
Niveau 3	5 tentatives	30 minutes	Oui (email automatique)

13.2 Dégel Automatique Intelligent

Déclencheur	Stratégie	Impact historique
Dégel niveau 1 → 2	UPDATE planification	Conservé
Dégel niveau 2 → 3	UPDATE planification	Conservé
Dégel niveau 3 complet	DELETE planification	Réinitialisé
Connexion réussie	DELETE planification	Réinitialisé
Dégel manuel admin	DELETE planification	Réinitialisé

13.3 Nouvelles Tables MySQL

- scheduled_unfreezes : planification et suivi des dégels automatiques
- password_histories : historique des mots de passe (réutilisation interdite sur 5 derniers)

13.4 Nouveaux Fichiers Frontend

- admin-dashboard.html : Interface d'administration dédiée
- assets/js/ynov-cron.js : Cron front-end avec intervalle adaptatif
- assets/js/password-expiry-checker.js : Vérificateur d'expiration de mot de passe

13.5 Nouveaux Endpoints

- GET /api/cron/check-unfreeze : Vérifier et dégeler un compte via login
- POST /api/cron/auto-unfreeze : Forcer le dégel d'un compte
- GET /api/cron/freeze-status/{uuid} : Obtenir le statut de gel détaillé
- GET /api/cron/frozen-accounts : Lister tous les comptes actuellement gelés
- GET /api/cron/check-unfreezes : Traiter les dégels en attente (batch)

13.6 Environnement de Production

Avant (v1.0)	Après (v2.0)

localhost:8000	https://apidev.yakoafricassur.com
localhost:5501	https://api-auth-ynov.yakoafricassur.com
Pas de doc API interactive	https://apidev.yakoafricassur.com/docs (accueil de l'api)
Email : ...@yakoafrica.ci	Email : ...@yakoafricassur.com

14. CONTACT & SUPPORT

14.1 Équipe de Développement

Rôle	Nom	Email
Chef de projet	Alexia AKE	alexia.ake@yakoafricassur.com
Lead Backend	Bruce YAPO	bruce.yapo@yakoafricassur.com
Lead Frontend	À définir	-

14.2 Support Technique

Canal	Coordonnée
Email	support@yakoafricassur.com
API Docs	https://apidev.yakoafricassur.com/docs (accueil de l'api)
Frontend	https://api-auth-ynov.yakoafricassur.com