



Plateforme READ

Guide d'exploitation

Installer, configurer, déployer, sauvegarder, sécuriser et surveiller la plateforme sur un serveur.

Public

Équipe technique (ICT) et hébergeur

Édition

Édition 1.0 — octobre 2026

Contenu

11 chapitres, 2 figures

Sommaire

	À propos de ce guide	5
	Ce que vous devez savoir avant de commencer	5
	Organisation du guide	5
	Conventions	6
Chapitre 1	Architecture	7
	1.1 Composants logiciels	7
	1.2 Conteneurs	8
	1.3 Volumes de données	8
	1.4 Flux réseau	9
	1.5 Organisation du serveur	9
	1.6 Tâches de fond	9
Chapitre 2	Prérequis	11
	2.1 Serveur	11
	2.2 Nom de domaine	11
	2.3 Messagerie (SMTP)	12
	2.4 Stockage hors serveur	12
	2.5 Poste de l'équipe technique	12
	2.6 Postes des utilisateurs	13
Chapitre 3	Installation	14
	3.1 Système	14
	3.2 Docker	15
	3.3 Fichier de réglages	15
	3.4 Premier déploiement	16
	3.5 Vérifications	16
	3.6 Après l'installation	17
Chapitre 4	Configuration	18
	4.1 Appliquer un changement	18
	4.2 Base de données	18
	4.3 Adresse du site	19
	4.4 Application et administrateur initial	19
	4.5 Messagerie	20
	4.6 Démonstration	20
	4.7 Sauvegarde	20
	4.8 Réglages avancés de l'application	21
Chapitre 5	Déploiement et mises à jour	23
	5.1 Principe	23
	5.2 Déployer une nouvelle version	23

	5.3	Migrations du schéma	24
	5.4	Revenir à la version précédente	24
	5.5	Versions figées	24
	5.6	Mise à jour mensuelle	25
Chapitre 6		Sauvegarde et restauration	26
	6.1	Ce qui est sauvegardé	26
	6.2	Activer la sauvegarde quotidienne	27
	6.3	Copie hors serveur chiffrée	27
	6.4	Restaurer	28
Chapitre 7		Sécurité	30
	7.1	Transport et en-têtes	30
	7.2	Comptes et sessions	30
	7.3	Autorisations et données	31
	7.4	Serveur	31
	7.5	Responsabilités de l'exploitant	31
	7.6	Signaler ou traiter une compromission	32
Chapitre 8		Supervision	33
	8.1	État des conteneurs	33
	8.2	Journaux	33
	8.3	Place disque	34
	8.4	Sauvegardes	34
	8.5	Alertes conseillées	35
	8.6	Statistiques d'utilisation	35
Chapitre 9		Dépannage	36
	9.1	Le certificat HTTPS n'est pas obtenu	36
	9.2	Le conteneur de l'application est « unhealthy » ou redémarre en boucle	36
	9.3	Les migrations échouent	37
	9.4	Le disque est plein	37
	9.5	Les e-mails ne partent pas	38
	9.6	L'heure affichée est fausse ou les connexions échouent par intermittence	38
	9.7	Le mot de passe de l'administrateur est perdu	38
	9.8	Trop de requêtes (429) ou blocage d'un bureau entier	39
	9.9	Toutes les adresses du journal d'audit sont identiques	39
	9.10	Une restauration d'essai échoue	39
	9.11	Les utilisateurs voient une ancienne version du module hors ligne	40
Chapitre 10		API et module hors ligne	41
	10.1	L'API REST	41
	10.2	Le module hors ligne	43
Chapitre 11		Environnement de démonstration	44
	11.1	Données de démonstration	44
	11.2	Remettre la démonstration à zéro	45

	11.3	Ce qu'il ne faut jamais faire sur une plateforme réelle	45
Annexe A		Aide-mémoire des commandes	46
	A.1	État et journaux	46
	A.2	Démarrer, arrêter, appliquer un réglage	46
	A.3	Déploiement (depuis le poste)	46
	A.4	Sauvegarde et restauration	46
	A.5	Base de données	47
	A.6	Fichiers et e-mails locaux	47
	A.7	Système et place disque	47
	A.8	Démonstration seulement	47

À propos de ce guide

À qui s'adresse ce guide, ce qu'il couvre et comment le lire.

Concerne : **Équipe technique** • **Hébergeur**

Ce guide décrit l'installation, la configuration, le déploiement, la sauvegarde, la sécurité et la surveillance de la plateforme READ sur un serveur. Il s'adresse à l'équipe technique (ICT) qui exploite la plateforme et à l'hébergeur.

L'utilisation fonctionnelle de la plateforme est décrite dans le [Guide de l'utilisateur](#) ; l'administration (comptes, paramètres, référentiels) dans le [Guide de l'administrateur](#).

Ce que vous devez savoir avant de commencer

Le guide suppose une pratique courante de :

- l'administration d'un serveur Linux (Ubuntu) en ligne de commande, par SSH ;
- Docker et Docker Compose ;
- les notions de nom de domaine, d'enregistrement DNS et de certificat HTTPS.

Aucune connaissance de .NET n'est nécessaire pour exploiter la plateforme : elle est livrée sous forme de conteneurs.

Organisation du guide

Chapitre	Contenu
Architecture	Composants, conteneurs, volumes, flux réseau
Prérequis	Serveur, nom de domaine, messagerie, stockage hors serveur
Installation	Préparation du serveur et premier déploiement
Configuration	Toutes les variables de réglage et leurs valeurs par défaut
Déploiement et mises à jour	Nouvelle version, retour arrière, migrations, mises à jour mensuelles
Sauvegarde et restauration	Sauvegarde quotidienne, copie chiffrée hors serveur, restauration
Sécurité	Mesures en place et responsabilités de l'exploitant
Supervision	Journaux, état de santé, place disque, alertes
Dépannage	Fiches d'incidents : symptôme, diagnostic, solution
API et module hors ligne	Jetons, durées, description OpenAPI, mise à jour du module
Environnement de démonstration	Données fictives et remise à zéro
Aide-mémoire	Les commandes utiles, regroupées

Conventions

- Les commandes sont à taper dans un terminal du serveur, sauf mention « depuis le poste » (le poste de l'équipe technique qui détient le code source).
- Les éléments à remplacer sont entre chevrons : `<alias-ssh>`, `<horodatage>`.
- Les exemples utilisent l'adresse fictive `read.example.org`.
- Les sorties de commande reproduites sont des **exemples** : les valeurs réelles diffèrent.

Important

Le fichier de réglages du serveur, `/opt/read/.env`, contient tous les secrets de la plateforme (mots de passe de la base, de l'administrateur initial, du serveur de messagerie, de la copie hors serveur). Il ne doit jamais être copié dans un ticket, un e-mail ou le dépôt de code.

CHAPITRE 1

Architecture

Les composants de la plateforme, les conteneurs qui les portent, les volumes de données et les flux réseau.

Concerne : **Équipe technique** • Hébergeur

La plateforme READ est une application web unique (un « monolithe modulaire »), livrée en conteneurs Docker et orchestrée par Docker Compose sur un seul serveur.

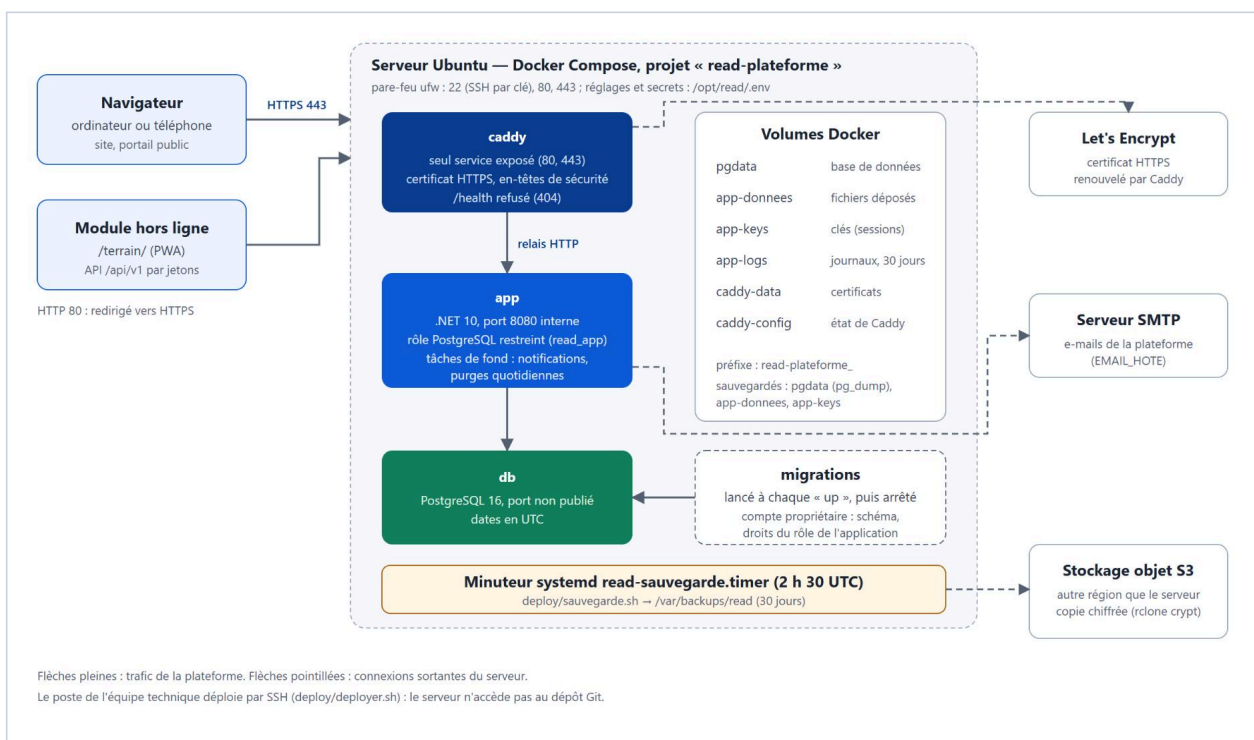


Figure 1.1 — Architecture de déploiement de la plateforme

1.1 Composants logiciels

- **Application web** en .NET 10 (Blazor, rendu côté serveur) : pages légères, peu de JavaScript (graphiques, calendrier, grille de saisie). Elle sert le site des comptes connectés (/admin, /espace), le portail public (/) et l'API (/api/v1).
- **Module hors ligne** (/terrain/) : application WebAssembly installable sur un téléphone (PWA). Elle est servie par la même application et dialogue avec la plateforme par l'API, au moyen de jetons.
- **Base de données** PostgreSQL 16. Les dates y sont enregistrées en UTC et affichées à l'heure de Niamey ; les montants sont en FCFA sans décimale.
- **Relais HTTPS Caddy** : il obtient et renouvelle automatiquement le certificat (Let's Encrypt), ajoute les en-têtes de sécurité, compresse les réponses et transmet les requêtes à l'application.

Chaque écriture en base produit une ligne du **journal d'audit** (anciennes et nouvelles valeurs), dans la même transaction. Le journal est en ajout seul : l'application n'a pas le droit de le modifier ni de le vider.

1.2 Conteneurs

Le projet Compose s'appelle toujours `read-plateforme` (fixé dans `compose.yaml`), quel que soit le dossier où se trouve le code. Les conteneurs portent donc des noms comme `read-plateforme-app-1`.

Service	Image	Rôle
db	postgres:16.15-alpine	Base de données ; port non publié sur l'hôte
migrations	read-plateforme:latest	Lancé à chaque <code>docker compose up</code> , puis arrêté : applique les migrations du schéma avec le compte propriétaire et crée ou met à jour le rôle restreint de l'application
app	read-plateforme:latest	Application, port 8080 interne ; se connecte avec le rôle restreint
caddy	caddy:2.11.7-alpine	Seul service exposé : ports 80 et 443 (TCP, et UDP pour HTTP/3)

L'image `read-plateforme:latest` est construite sur le serveur à partir du code source (`Dockerfile`) : aucun registre d'images n'est nécessaire.

Remarque

Au premier démarrage sur une base vide, le service migrations écrit dans son journal une erreur sur la table `__EFMigrationsHistory` avant de la créer. Elle est normale.

1.3 Volumes de données

Volume	Contenu	Sauvegardé
read-plateforme_pgdata	Fichiers de la base PostgreSQL	Oui, par <code>pg_dump</code>
read-plateforme_app-donnees	Fichiers déposés : logo, pièces jointes, ressources de la bibliothèque, e-mails écrits localement (boîte-envoi/)	Oui
read-plateforme_app-keys	Clés de protection des données (cookies de session, jetons)	Oui
read-plateforme_app-logs	Journaux quotidiens de l'application, 30 jours	Non
read-plateforme_caddy-data	Certificats HTTPS	Non (redemandés automatiquement)
read-plateforme_caddy-config	État de Caddy	Non

Attention

Ne supprimez jamais les volumes `pgdata`, `app-donnees` ou `app-keys` : `docker compose down -v` efface toutes les données de la plateforme. Sans `app-keys`, toutes les sessions et tous les jetons du module hors ligne deviennent invalides (les utilisateurs doivent se reconnecter).

1.4 Flux réseau

Sens	Depuis	Vers	Port	Usage
Entrant	Internet	<code>caddy</code>	80/TCP	Redirection vers HTTPS, validation du certificat
Entrant	Internet	<code>caddy</code>	443/TCP et 443/UDP	Site, portail, API, module hors ligne
Entrant	Poste de l'équipe technique	Serveur	22/TCP	SSH (déploiement, exploitation)
Interne	<code>caddy</code>	<code>app</code>	8080	Relais HTTP sur le réseau Docker
Interne	<code>app, migrations</code>	<code>db</code>	5432	Base de données
Sortant	<code>caddy</code>	Let's Encrypt	443	Obtention et renouvellement du certificat
Sortant	<code>app</code>	Serveur SMTP	587 ou 465 (réglable)	E-mails (mot de passe oublié, notifications)
Sortant	Sauvegarde	Stockage objet S3	443	Copie chiffrée hors serveur

L'application ne croit l'adresse du client transmise par Caddy (en-tête `X-Forwarded-For`) que si la requête vient d'un réseau privé (réseau Docker) : la limitation des connexions et le journal d'audit voient la vraie adresse du visiteur.

1.5 Organisation du serveur

Emplacement	Contenu
<code>/opt/read/.env</code>	Réglages et secrets (droits 600) ; lié dans <code>code/deploy/.env</code>
<code>/opt/read/code</code>	Code de la version déployée ; fichier <code>VERSION</code> = commit
<code>/opt/read/code.ancien</code>	Version précédente, pour un retour arrière
<code>/opt/read/deploiement.log</code>	Compte rendu du dernier déploiement
<code>/var/backups/read</code>	Sauvegardes locales, un dossier horodaté par jour
<code>/etc/systemd/system/read-sauvegarde.*</code>	Minuteur de la sauvegarde quotidienne

1.6 Tâches de fond

L'application exécute elle-même quelques tâches, sans planificateur externe :

- **chaque minute** : envoi des e-mails de notification en attente ;
- **chaque jour à partir de 7 h (heure de Niamey)** : rappels (activités et rapports en retard), puis suppression des notifications de plus de 90 jours ;
- **chaque jour** (première passe cinq minutes après le démarrage) : suppression des listes de présence arrivées à échéance, effacement des coordonnées des demandes d'inscription refusées depuis 12 mois, purge des jetons de l'API expirés ou révoqués.

La sauvegarde, elle, est lancée par un minuteur systemd du serveur (chapitre [Sauvegarde et restauration](#)).

CHAPITRE 2

Prérequis

Ce qu'il faut réunir avant l'installation : serveur, nom de domaine, messagerie, stockage hors serveur et postes des utilisateurs.

Concerne : **Équipe technique** • Hébergeur

2.1 Serveur

Configuration validée (serveur de démonstration) :

Élément	Minimum conseillé
Système	Ubuntu 26.04 LTS (64 bits)
Processeur	2 vCPU
Mémoire	4 Go, plus 2 Go d'espace d'échange (swap)
Disque	40 Go (SSD)
Réseau	Adresse IPv4 publique (IPv6 conseillée), ports 80 et 443 joignables depuis Internet
Accès	SSH par clé, compte d'exploitation avec sudo

Important

La construction de l'image .NET a lieu sur le serveur : sous 4 Go de mémoire sans espace d'échange, elle peut échouer faute de mémoire. Créez le swap de 2 Go décrit à l'[installation](#).

Place disque. Elle dépend surtout des fichiers déposés (photos, pièces jointes de 10 Mo au plus) et des 30 jours de sauvegardes locales : chaque sauvegarde contient la base et **tous** les fichiers déposés. Prévoyez une marge d'au moins trois fois `30 × (taille de la base + taille des fichiers)` sur la durée du projet, et surveillez la place (chapitre [Supervision](#)).

Logiciels installés sur le serveur : Docker Engine et le greffon Docker Compose, depuis le dépôt officiel de Docker. Rien d'autre : PostgreSQL, .NET, Caddy, rclone et les outils de sauvegarde tournent dans des conteneurs.

2.2 Nom de domaine

La plateforme doit être servie sur un nom de domaine (par exemple `read.example.org`) pour obtenir un certificat HTTPS.

1. Choisissez le nom (sous-domaine d'un domaine existant de l'organisation, de préférence).
2. Créez chez le gestionnaire DNS un enregistrement A vers l'adresse IPv4 du serveur et, si le serveur en a une, un enregistrement AAAA vers son adresse IPv6.

3. Attendez que le nom résolve vers le serveur avant le premier démarrage de Caddy.

Attention

Si le domaine passe par un service de type « proxy » ou CDN (par exemple le nuage orange de Cloudflare), désactivez ce mode pour cet enregistrement (« DNS seulement »). Sinon la plateforme voit l'adresse du service au lieu de celle des visiteurs : la limitation des tentatives de connexion et le journal d'audit deviennent faux.

En attendant un domaine, le nom fourni par l'hébergeur pour le serveur (de la forme `vps-...example.net`) convient : Caddy obtient aussi un certificat pour lui. Le jour où le domaine est prêt, passez `SITE_ADDRESS` et `URL_PUBLIQUE` sur le domaine et mettez l'ancien nom dans `SITE_REDIRECTION` : les liens déjà diffusés continuent de fonctionner.

2.3 Messagerie (SMTP)

La plateforme envoie des e-mails : lien « mot de passe oublié », copies des notifications. Il faut un compte sur un serveur SMTP (serveur de l'organisation ou service d'envoi transactionnel) : hôte, port, mode de chiffrement, identifiant, mot de passe et adresse d'expédition autorisée.

Sans serveur SMTP, aucun e-mail ne part : les messages sont écrits dans le volume des fichiers déposés (`boîte-envoi/`), ce qui convient pour un essai mais pas en production.

Remarque

L'envoi passe par une authentification SMTP classique (identifiant et mot de passe). Un fournisseur qui n'accepte que OAuth (par exemple certaines configurations de Microsoft 365) demande une adaptation du code.

2.4 Stockage hors serveur

Le cahier des charges exige une copie des sauvegardes **hors du serveur**. Il faut un stockage objet compatible S3, situé dans une **autre région** que le serveur (par exemple OVHcloud Object Storage), avec :

- un conteneur (« bucket ») dédié ;
- un utilisateur S3 limité à ce conteneur : point d'accès, région, clé d'accès, clé secrète ;
- une phrase de chiffrement longue, conservée **hors du serveur** (gestionnaire de mots de passe de l'organisation).

La procédure figure au chapitre [Sauvegarde et restauration](#).

2.5 Poste de l'équipe technique

Le déploiement se lance depuis un poste qui dispose :

- du dépôt Git du code source ;
- de Git et d'un shell bash (Linux, macOS, ou Git Bash sous Windows) ;
- d'un client SSH dont la clé est autorisée sur le serveur, et d'un alias SSH (fichier `~/.ssh/config`) vers le

serveur.

Le serveur, lui, n'a besoin d'aucun accès au dépôt.

2.6 Postes des utilisateurs

- **Navigateurs** : versions récentes de Chrome, Edge, Firefox ou Safari, sur ordinateur ou téléphone. JavaScript activé (graphiques, calendrier, grille de saisie).
- **Écran** : chaque page est utilisable à partir de 360 pixels de large (téléphone).
- **Module hors ligne** : téléphone Android avec Chrome (installation « Ajouter à l'écran d'accueil »). Une première connexion avec du réseau est nécessaire ; la saisie se fait ensuite sans réseau.
- **Bande passante** : les pages sont légères ; le module hors ligne télécharge son application (quelques mégaoctets) une seule fois, puis à chaque nouvelle version.

CHAPITRE 3

Installation

Préparer un serveur Ubuntu, créer le fichier de réglages et réaliser le premier déploiement.

Concerne : **Équipe technique** • Hébergeur

Ce chapitre installe la plateforme sur un serveur neuf. Comptez une heure, dont une dizaine de minutes pour la première construction de l'image.

3.1 Système

Connectez-vous au serveur avec le compte d'exploitation (ici `ubuntu`), puis :

1. Limitez SSH aux clés et interdisez la connexion directe de `root`. Le fichier est lu avant `50-cloud-init.conf` : la première valeur rencontrée l'emporte.

```
sudo tee /etc/ssh/sshd_config.d/10-read.conf <<'EOF'
PasswordAuthentication no
KbdInteractiveAuthentication no
PermitRootLogin no
EOF
sudo sshd -t && sudo systemctl reload ssh
```

2. Ouvrez le pare-feu pour SSH, HTTP et HTTPS (HTTP/3 compris) :

```
sudo ufw allow OpenSSH && sudo ufw allow 80/tcp && sudo ufw allow 443/tcp && sudo ufw allow 443/udp
sudo ufw --force enable
```

3. Créez un espace d'échange de 2 Go :

```
sudo fallocate -l 2G /swapfile && sudo chmod 600 /swapfile && sudo mkswap /swapfile && sudo swapon /swapfile
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

4. Vérifiez que les mises à jour de sécurité automatiques sont actives :

```
systemctl is-enabled unattended-upgrades
```

5. Vérifiez que l'horloge est synchronisée (System clock synchronized: yes):

```
timedatectl
```

Attention

Gardez une session SSH ouverte pendant que vous modifiez SSH et le pare-feu, et testez une nouvelle connexion avant de la fermer : une erreur pourrait vous enfermer dehors.

Remarque

Docker publie ses ports en contournant `ufw`. Seuls les ports 80 et 443 de Caddy sont publiés : la base et l'application ne sont pas joignables de l'extérieur.

3.2 Docker

1. Installez Docker Engine depuis le dépôt officiel (<https://download.docker.com/linux/ubuntu>), en suivant la procédure de docs.docker.com, avec les paquets `docker-ce` `docker-ce-cli` `containerd.io` `docker-buildx-plugin` `docker-compose-plugin`.
2. Autorisez le compte d'exploitation à utiliser Docker et créez les dossiers de la plateforme :

```
sudo usermod -aG docker ubuntu
sudo mkdir -p /opt/read && sudo chown ubuntu:ubuntu /opt/read
sudo mkdir -p /var/backups/read && sudo chmod 700 /var/backups/read
```

3. Déconnectez-vous puis reconnectez-vous pour que l'appartenance au groupe `docker` prenne effet, et vérifiez :

```
docker version && docker compose version
```

3.3 Fichier de réglages

Le fichier `/opt/read/.env` porte tous les réglages et secrets du serveur. Il est créé une fois, à partir du modèle `deploy/.env.example` du dépôt ; chaque variable est décrite au chapitre [Configuration](#).

1. Copiez le contenu de `deploy/.env.example` dans `/opt/read/.env` et protégez le fichier :

```
chmod 600 /opt/read/.env
```

2. Générez un mot de passe différent pour `POSTGRES_PASSWORD`, `POSTGRES_APP_PASSWORD` et `ADMIN_MOT_DE_PASSE`, par exemple :

```
openssl rand -base64 48 | tr -dc 'A-Za-z0-9' | head -c 32; echo
```

Le mot de passe de l'administrateur doit compter au moins 8 caractères dont un chiffre.

3. Réglez au minimum :

Variable	Valeur
SITE_ADDRESS	read.example.org (le nom de domaine, sans https://)
URL_PUBLIQUE	https://read.example.org
ADMIN_EMAIL	Adresse de la personne qui administrera la plateforme
ASPNETCORE_ENVIRONMENT	Production
DEMO_AFFICHER, DEMO_DONNEES	false
EMAIL_*	Le serveur SMTP

Danger

Sur une plateforme réelle, DEMO_DONNEES doit valoir false : à true, des données entièrement fictives seraient chargées dans une base vide et mêlées ensuite aux vraies.

3.4 Premier déploiement

Depuis le poste de l'équipe technique, à la racine du dépôt, sur le commit à déployer :

```
deploy/deployer.sh <alias-ssh>
```

Sous Windows (Git Bash), passez par le client OpenSSH du système, qui voit l'agent de clés :

```
SSH=/c/Windows/System32/OpenSSH/ssh.exe deploy/deployer.sh <alias-ssh>
```

Le script envoie le code du commit courant dans /opt/read/code, lie /opt/read/.env, puis lance la construction et le démarrage en arrière-plan. La première construction télécharge le SDK .NET et les paquets NuGet (une dizaine de minutes). Suivez-la sur le serveur :

```
tail -f /opt/read/déploiement.log
```

Le détail du déploiement est au chapitre [Déploiement et mises à jour](#).

3.5 Vérifications

1. Les conteneurs sont en place : app « healthy », migrations « Exited (0) ».

```
cd /opt/read/code && docker compose ps -a
```

2. Le site répond en HTTPS sur https://read.example.org, et http://read.example.org redirige vers HTTPS.
3. L'état de santé n'est pas public : https://read.example.org/health répond 404.
4. L'administrateur se connecte avec ADMIN_EMAIL et ADMIN_MOT_DE_PASSE ; la plateforme lui demande aussitôt de choisir un nouveau mot de passe.
5. Videz ensuite ADMIN_MOT_DE_PASSE dans /opt/read/.env : il ne sert qu'à la création du premier

administrateur.

6. Essayez l'envoi d'un e-mail : « Mot de passe oublié » sur la page de connexion, pour le compte administrateur.

3.6 Après l'installation

- Activez la sauvegarde quotidienne et la copie hors serveur ([Sauvegarde et restauration](#)), puis faites une restauration d'essai.
- Mettez en place une surveillance de disponibilité ([Supervision](#)).
- Confiez la suite à l'administrateur fonctionnel : paramètres, logo, comptes, référentiels ([Guide de l'administrateur](#)).

CHAPITRE 4

Configuration

Toutes les variables du fichier de réglages et les réglages avancés de l'application, avec leur valeur par défaut et leur effet.

Concerne : **Équipe technique**

La plateforme se règle à deux niveaux :

- le fichier `/opt/read/.env` du serveur (modèle : `deploy/.env.example`), lu par Docker Compose ;
- des **réglages avancés** de l'application, rarement utiles, au format `Section__Cle`.

Les réglages fonctionnels (seuil de masquage du portail, saisie par les organisations, conservation des listes de présence, logo) ne sont pas ici : l'administrateur les change dans l'écran **Paramètres** de la plateforme, sans redémarrage ([Guide de l'administrateur](#)).

4.1 Appliquer un changement

Après une modification de `/opt/read/.env`, recréez les conteneurs concernés :

```
cd /opt/read/code
docker compose up -d
```

Compose recrée seulement les services dont la configuration a changé. Une reconstruction de l'image n'est pas nécessaire.

Important

`.env` suit la syntaxe de Docker Compose, pas celle de bash : une ligne par variable, `CLE=valeur`, sans espace autour du signe égal. Les scripts du dossier `deploy/` lisent les clés une à une ; ne « sourcez » jamais ce fichier dans un shell (une valeur comme `Plateforme READ y` serait exécutée).

4.2 Base de données

Variable	Défaut	Effet
<code>POSTGRES_DB</code>	<code>read</code>	Nom de la base
<code>POSTGRES_USER</code>	<code>read</code>	Compte propriétaire de la base. Il ne sert qu'au service migrations et aux scripts de sauvegarde et de restauration
<code>POSTGRES_PASSWORD</code>	aucun (obligatoire)	Mot de passe du compte propriétaire. Fixé à la création du volume <code>pgdata</code> : le changer ensuite dans <code>.env</code> ne change pas le mot de passe en base
<code>POSTGRES_APP_USER</code>	<code>read_app</code>	Rôle de connexion de l'application : lecture et écriture des

		tables, journal d'audit en ajout seul, aucun droit de propriétaire. Créé par le service migrations
POSTGRES_APP_PASSWORD	aucun (obligatoire)	Mot de passe de ce rôle, différent du précédent. Réappliqué à chaque <code>docker compose up</code> : il peut être changé librement

Sans `POSTGRES_PASSWORD` ou `POSTGRES_APP_PASSWORD`, `docker compose up` s'arrête avec un message explicite.

Attention

Pour changer `POSTGRES_PASSWORD` après l'installation, changez d'abord le mot de passe dans PostgreSQL (`docker compose exec db psql -U read -d read -c "ALTER ROLE read PASSWORD '...'";`), puis dans `.env`, puis `docker compose up -d`.

4.3 Adresse du site

Variable	Défaut	Effet
SITE_ADDRESS	:80	Adresse servie par Caddy. Un nom de domaine (<code>read.example.org</code>) active le HTTPS automatique et la redirection de HTTP vers HTTPS ; :80 sert toute adresse en HTTP simple (poste local seulement)
SITE_REDIRECTION	(vide)	Ancien nom à rediriger définitivement vers <code>URL_PUBLIQUE</code> , chemin et paramètres compris, après un changement de domaine (par exemple le nom fourni par l'hébergeur) ; Caddy garde un certificat pour lui. Vide : aucune redirection
HTTP_PORT	80	Port HTTP publié sur l'hôte
HTTPS_PORT	443	Port HTTPS publié sur l'hôte (TCP et UDP)
URL_PUBLIQUE	<code>http://localhost:8088</code>	Adresse du site écrite dans les e-mails (lien « mot de passe oublié »). Jamais déduite de la requête, pour qu'un lien ne puisse pas être détourné : à régler sur <code>https://<nom de domaine></code>

En production, gardez 80 et 443 : Let's Encrypt valide le domaine sur ces ports.

4.4 Application et administrateur initial

Variable	Défaut	Effet
ASPNETCORE_ENVIRONMENT	Production	Production sur un serveur : pages d'erreur sans détail technique, journaux quotidiens en fichiers, HSTS
ADMIN_EMAIL	vide	Adresse du premier administrateur
ADMIN_MOT_DE_PASSE	vide	Son mot de passe (8 caractères au moins, dont un chiffre). Le compte n'est créé que si aucun administrateur n'existe ; ensuite la variable est ignorée : videz-la
ADMIN_CHANGEMENT_OBLIGATOIRE	true	Oblige ce premier administrateur à changer son mot de passe à la première connexion. false seulement pour la

d mo en ligne

4.5 Messagerie

Variable	D�faut	Effet
EMAIL_HOTE	vide	Serveur SMTP. Vide : aucun e-mail ne part, chaque message est �crit en fichier .eml dans le dossier <code>bo�te-envoi/</code> du volume <code>app-donnees</code>
EMAIL_PORT	587	Port SMTP
EMAIL_SECURITE	Auto	Auto (STARTTLS si le serveur le propose), StartTls, SslOnConnect (port 465) ou None
EMAIL_UTILISATEUR	vide	Identifiant SMTP ; vide : pas d'authentification
EMAIL_MOT_DE_PASSE	vide	Mot de passe SMTP
EMAIL_EXPEDITEUR	ne-pas-repondre@read.local	Adresse d'exp�dition ; elle doit �tre autoris�e par le serveur SMTP (et par les enregistrements SPF/DKIM du domaine)
EMAIL_NOM_EXPEDITEUR	Plateforme READ	Nom affich� de l'exp�diteur

4.6 D monstration

  laisser   `false` et vide sur une plateforme r elle (chapitre [Environnement de d monstration](#)).

Variable	D�faut	Effet
DEMO_AFFICHER	false	Bandeau « Donn�es de d�monstration fictives » en haut de chaque page
DEMO_DONNEES	false	Charge au d�marrage des donn�es enti�rement fictives, seulement dans une base qui ne contient aucune organisation
DEMO_MOT_DE_PASSE	vide	Mot de passe des comptes de d�monstration ; sans lui, rien n'est charg�
DEMO_COMPTEES_PROTEGES	false	Comptes ...@demo.read verrouill�s : mot de passe inchangeable, ni modification ni d�sactivation par l'administrateur (d�mo en ligne seulement)

4.7 Sauvegarde

Lues par `deploy/sauvegarde.sh` (chapitre [Sauvegarde et restauration](#)), pas par l'application.

Variable	D�faut	Effet
SAUVEGARDE_DOSSIER	/var/backups/read	Dossier des sauvegardes locales
SAUVEGARDE_JOURS	30	Conservation, en jours, locale et distante
SAUVEGARDE_DISTANTE	vide	Destination rclone de la copie hors serveur, par exemple <code>chiffre:</code> . Vide : pas de copie (le compte rendu l'indique)

RCLONE_CONFIG_*	aucune	Description des destinations rclone (stockage S3 et couche de chiffrement) ; modèle commenté dans <code>deploy/.env.example</code>
-----------------	--------	--

4.8 Réglages avancés de l'application

Ces réglages ont une valeur par défaut adaptée. Ils ne sont **pas** transmis par `.env` : pour en changer un, ajoutez la variable dans la section `environment` du service `app` de `deploy/docker-compose.yml` (dans le dépôt, pour qu'elle survive aux déploiements), puis déployez.

```
# Exemple (deploy/docker-compose.yml, service app)
environment:
  Api__DureeJetonAccesMinutes: "30"
```

Réglage	Défaut	Effet
Securite__ConnexionsParMinute	10	Tentatives de connexion par minute et par adresse IP (site et jetons de l'API) ; au-delà, réponse 429
Securite__InscriptionsParHeure	5	Demandes d'inscription d'organisation par heure et par adresse IP
Securite__ReinitialisationsParHeure	5	Demandes « mot de passe oublié » par heure et par adresse IP
Securite__ReseauxProxy__0,__1...	réseaux privés	Réseaux (CIDR) dont l'en-tête X-Forwarded-For est cru. Par défaut 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, fc00::/7 (réseau Docker)
Securite__PolitiqueContenuRapportSeulement	false	true : la politique de sécurité du contenu (CSP) est seulement signalée par le navigateur, sans rien bloquer. Repli de dépannage, à ne pas laisser en service
Securite__IntervalleRevalidationSecondes	60	Délai au bout duquel une session ouverte voit un compte désactivé ou un rôle changé
Api__DureeJetonAccesMinutes	30	Durée d'un jeton d'accès de l'API
Api__DureeRenouvellementJours	14	Durée glissante du jeton de renouvellement (module hors ligne)
Api__DureeMaximaleJours	90	Au-delà, le mot de passe est redemandé sur l'appareil
Api__DelaiRejeuSecondes	120	Rejeu toléré d'un jeton de renouvellement tout juste utilisé (réseau instable)
Api__AppelsParMinute	300	Appels à l'API par minute et par adresse IP
Notifications__Auto	true	false : arrête l'envoi des e-mails de notification, les rappels et la purge des notifications
PiecesJointes__PurgeAuto	true	false : arrête les tâches quotidiennes de conservation (listes de présence échues, demandes refusées, jetons expirés)

Serilog__MinimumLevel__Default	Informati on	Niveau des journaux (Debug pour un diagnostic ponctuel)
--------------------------------	-----------------	---

Réglages fixés par l'image ou par `deploy/docker-compose.yml`, à ne pas modifier :

`ConnectionStrings__Read` (construite à partir des variables `POSTGRES_*`), `Base__Migrer` (`false` : l'application ne migre pas elle-même, le service `migrations` le fait), `Base__MigrerSeulement` (service `migrations`), `Base__RoleApplication` et `Base__MotDePasseRoleApplication` (rôle restreint), `Base__InitialiserAuDemarrage` (`true`), `Stockage__Dossier` (`donnees`, soit `/app/donnees` dans le conteneur), `DataProtection__KeysPath` (`/app/keys`), `ASPNETCORE_HTTP_PORTS` (`8080`).

Danger

Les tâches de conservation (`PiecesJointes__PurgeAuto`) appliquent des engagements de protection des données (listes de présence supprimées à échéance, coordonnées effacées). Ne les désactivez que le temps d'un diagnostic.

CHAPITRE 5

Déploiement et mises à jour

Déployer une nouvelle version, revenir à la précédente, comprendre les migrations et tenir à jour les composants chaque mois.

Concerne : **Équipe technique**

5.1 Principe

Le déploiement part toujours du **poste de l'équipe technique**, qui détient le dépôt Git. Le script `deploy/deployer.sh` :

1. signale les fichiers suivis modifiés mais non commités (ils ne seront pas déployés) ;
2. envoie au serveur, par SSH, l'archive du **commit courant** (`git archive HEAD`) : jamais un fichier non commité, jamais `deploy/.env` ;
3. sur le serveur, décompresse le code dans `/opt/read/code.nouveau`, y lie `/opt/read/.env` et écrit le commit dans `VERSION` ;
4. remplace `/opt/read/code` par la nouvelle version et garde la précédente dans `/opt/read/code.ancien` ;
5. lance `docker compose up -d --build` en arrière-plan, avec le compte rendu dans `/opt/read/déploiement.log`.

Le serveur n'a besoin d'aucun accès au dépôt Git.

5.2 Déployer une nouvelle version

1. Sur le poste, placez-vous sur le commit à déployer (branche `main` à jour, tests réussis).
2. Lancez :

```
deploy/deployer.sh <alias-ssh>
```

Sous Windows : `SSH=/c/Windows/System32/OpenSSH/ssh.exe deploy/deployer.sh <alias-ssh>`.

3. Suivez la construction sur le serveur :

```
tail -f /opt/read/déploiement.log
```

Les constructions suivantes réutilisent les paquets NuGet déjà téléchargés (cache de construction) : quelques minutes suffisent.

4. Vérifiez la version et l'état :

```
cat /opt/read/code/VERSION
cd /opt/read/code && docker compose ps -a
```

Attendu : app « healthy », migrations « Exited (0) », db et caddy « Up ».

5. Ouvrez le site, connectez-vous, parcourez quelques pages.

Pendant la construction, l'ancienne version reste en service ; seul le remplacement des conteneurs provoque une coupure de quelques secondes.

Conseil

Faites une sauvegarde juste avant un déploiement qui change le schéma de la base (`sudo /opt/read/code/deploy/sauvegarde.sh`) : c'est elle qui permettra un retour arrière complet.

5.3 Migrations du schéma

À chaque `docker compose up`, le service `migrations` démarre avant l'application :

- il applique les migrations du schéma manquantes, avec le compte propriétaire (`POSTGRES_USER`) ;
- il crée ou met à jour le rôle de l'application (`POSTGRES_APP_USER`) : droits de lecture et d'écriture sur toutes les tables (y compris les nouvelles), mais ni modification ni effacement du journal d'audit, ni écriture dans l'historique des migrations ;
- il s'arrête. L'application ne démarre que s'il s'est terminé sans erreur.

L'application ne migre jamais elle-même en production (`Base__Migrer=false`).

5.4 Revenir à la version précédente

Si la nouvelle version pose problème, sur le serveur :

```
cd /opt/read && rm -rf code && mv code.ancien code && cd code && docker compose up -d --build
```

Attention

Une migration déjà appliquée n'est pas défaite par ce retour arrière. Si la version fautive a modifié le schéma, l'ancienne version peut ne plus fonctionner avec la base : restaurez alors la sauvegarde faite avant le déploiement ([Restauration](#)).

Un seul niveau de retour arrière est conservé sur le serveur. Pour revenir plus loin, déployez l'ancien commit depuis le poste (`git checkout <commit>` puis `deploy/deployer.sh`).

5.5 Versions figées

Pour qu'une construction soit reproductible, toutes les images et outils ont une version exacte :

Composant	Version	Fichier
SDK .NET (construction)	mcr.microsoft.com/dotnet/sdk: 10.0.401	Dockerfile
ASP.NET (exécution)	mcr.microsoft.com/dotnet/aspnet: 10.0.12	Dockerfile

PostgreSQL	postgres:16.15-alpine	deploy/docker-compose.yml, deploy/restaurer.sh
Caddy	caddy:2.11.7-alpine	deploy/docker-compose.yml
Alpine (outils de sauvegarde)	alpine:3.24	deploy/sauvegarde.sh, deploy/restaurer.sh, deploy/reinitialiser-demo.sh
rclone	rclone/rclone:1.75.1	deploy/sauvegarde.sh
Paquets NuGet	versions exactes	Directory.Packages.props

5.6 Mise à jour mensuelle

Chaque mois, pour recevoir les correctifs de sécurité :

1. Relevez la dernière version **de la même branche** de chaque image : Docker Hub pour postgres:16.x-alpine, caddy:2.x-alpine, alpine et rclone/rclone ; la liste <https://mcr.microsoft.com/v2/dotnet/aspnet/tags/list> (et .../dotnet/sdk/...) pour .NET 10.
2. Mettez à jour les fichiers du tableau ci-dessus.
3. Lancez la suite de tests (dotnet test, Docker requis), committez.
4. Faites une sauvegarde, puis déployez.
5. Vérifiez le site, puis lancez une restauration d'essai.

Le système du serveur se met à jour seul (unattended-upgrades). Docker, lui, se met à jour par apt : `sudo apt update && sudo apt upgrade` lors d'une fenêtre de maintenance, suivi d'une vérification des conteneurs.

Danger

Ne changez jamais la version **majeure** de PostgreSQL (16) en modifiant seulement l'image : les fichiers du volume pgdata ne sont pas compatibles d'une version majeure à l'autre. Une montée de version se fait par une sauvegarde (pg_dump), une base neuve et une restauration.

CHAPITRE 6

Sauvegarde et restauration

La sauvegarde quotidienne, la copie chiffrée hors serveur, la restauration d'essai et la restauration réelle.

Concerne : Équipe technique • Hébergeur

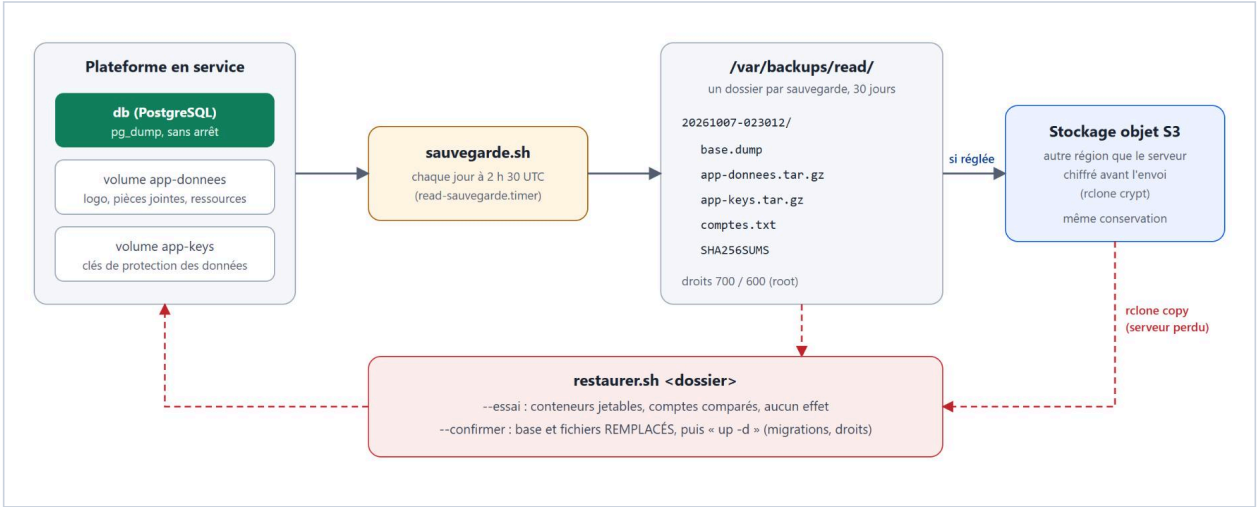


Figure 6.1 — Sauvegarde quotidienne, copie hors serveur et restauration

6.1 Ce qui est sauvegardé

Le script `deploy/sauvegarde.sh` produit, sans arrêter la plateforme, un dossier horodaté (heure UTC) par sauvegarde dans `SAUEGARDE_DOSSIER (/var/backups/read)` :

Fichier	Contenu
<code>base.dump</code>	Base PostgreSQL complète (<code>pg_dump</code> , format personnalisé), sans propriétaires ni droits : un instantané cohérent même pendant l'utilisation
<code>app-donnees.tar.gz</code>	Fichiers déposés : logo, pièces jointes, ressources de la bibliothèque
<code>app-keys.tar.gz</code>	Clés de protection des données ; sans elles, les sessions et les jetons émis deviennent illisibles
<code>comptes.txt</code>	Nombres de lignes de quelques tables (organisations, activités, rapports, participations, pièces jointes, utilisateurs, journal d'audit) et nombres de fichiers ; comparés lors d'une restauration
<code>SHA256SUMS</code>	Empreintes des quatre fichiers précédents

Les fichiers sont lisibles par `root` seulement (droits 600, dossier 700). Les sauvegardes de plus de `SAUEGARDE_JOURS` jours (30) sont supprimées à chaque exécution, localement et à distance.

Remarque

Les journaux de l'application et les certificats HTTPS ne sont pas sauvegardés : les journaux ne sont pas des données métier et Caddy redemande les certificats.

6.2 Activer la sauvegarde quotidienne

Le minuteur systemd fourni lance la sauvegarde chaque jour à 2 h 30 UTC (3 h 30 à Niamey), à quelques minutes près ; une exécution manquée (serveur éteint) est rattrapée au démarrage.

```
sudo cp /opt/read/code/deploy/systemd/read-sauvegarde.* /etc/systemd/system/  
sudo systemctl daemon-reload && sudo systemctl enable --now read-sauvegarde.timer  
systemctl list-timers read-sauvegarde.timer          # prochaine exécution
```

Lancer une sauvegarde à la main (avant un déploiement, par exemple) :

```
sudo /opt/read/code/deploy/sauvegarde.sh
```

Lire le compte rendu de la dernière sauvegarde automatique :

```
sudo journalctl -u read-sauvegarde.service -n 20
```

Exemple de compte rendu (valeurs indicatives) :

```
[2026-10-08T02:31:04Z] Sauvegarde dans /var/backups/read/20261008-023104  
[2026-10-08T02:31:19Z] Taille : 48M ; organisations 24;activites 300;...;fichiers_app-keys 1;  
[2026-10-08T02:31:41Z] Copie hors serveur : chiffre:/20261008-023104  
[2026-10-08T02:31:42Z] Sauvegarde terminée.
```

Attention

Le message ATTENTION : aucune copie hors serveur (SAUVEGARDE_DISTANTE vide) signifie que les sauvegardes restent sur le serveur : elles disparaîtraient avec lui. Configurez la copie hors serveur.

6.3 Copie hors serveur chiffrée

La copie est faite par rclone (dans un conteneur : rien à installer), **chiffrée avant l'envoi** (couche `crypt`) : l'hébergeur du stockage ne peut lire ni les contacts ni le journal.

1. Créez un stockage objet compatible S3 **dans une autre région que le serveur**, un conteneur et un utilisateur S3 limité à ce conteneur. Notez le point d'accès, la région, la clé d'accès et la clé secrète.
2. Choisissez une phrase de chiffrement longue et **conservez-la hors du serveur** (gestionnaire de mots de passe). Convertissez-la au format attendu par rclone :

```
docker run --rm rclone/rclone:1.75.1 obscure '<phrase>'
```

3. Ajoutez à `/opt/read/.env` (modèle commenté dans `deploy/.env.example`) :

```
RCLONE_CONFIG_DISTANT_TYPE=s3
RCLONE_CONFIG_DISTANT_PROVIDER=other
RCLONE_CONFIG_DISTANT_ENDPOINT=https://s3.<region>.example.net
RCLONE_CONFIG_DISTANT_REGION=<region>
RCLONE_CONFIG_DISTANT_ACCESS_KEY_ID=<clé d'accès>
RCLONE_CONFIG_DISTANT_SECRET_ACCESS_KEY=<clé secrète>
RCLONE_CONFIG_CHIFFRE_TYPE=crypt
RCLONE_CONFIG_CHIFFRE_REMOTE=distant:<conteneur>/read
RCLONE_CONFIG_CHIFFRE_PASSWORD=<phrase convertie par « rclone obscure »>
SAUEGARDE_DISTANTE=chiffre:
```

4. Lancez une sauvegarde à la main ; le compte rendu doit indiquer Copie hors serveur.
5. Listez les copies distantes :

```
docker run --rm --env-file <(grep ^RCLONE_ /opt/read/.env) rclone/rclone:1.75.1 lsd chiffre:
```

Danger

Sans la phrase de chiffrement, les copies distantes sont illisibles, y compris pour vous. Si le serveur est perdu, c'est la copie conservée hors du serveur qui permet de tout restaurer : vérifiez qu'au moins deux personnes savent où elle se trouve.

6.4 Restaurer

Le script `deploy/restaurer.sh` vérifie d'abord les empreintes (SHA256SUMS) du dossier fourni, puis travaille selon le mode choisi.

6.4.1 Restauration d'essai

Elle restaure la sauvegarde dans des conteneurs **jetable**s et compare les nombres de lignes et de fichiers avec ceux relevés à la sauvegarde. Elle n'a **aucun effet** sur la plateforme.

```
sudo /opt/read/code/deploy/restaurer.sh /var/backups/read/<horodatage> --essai
```

Elle se termine par Restauration d'essai réussie, identique à la sauvegarde, ou par ÉCHEC et les différences.

Important

Faites une restauration d'essai au moins une fois par mois et après chaque changement du schéma de la base. Une sauvegarde jamais relue n'est pas une sauvegarde.

6.4.2 Restauration réelle

Elle **remplace** la base et les fichiers déposés de la plateforme par ceux de la sauvegarde.

1. Prévenez les utilisateurs : tout ce qui a été saisi après la sauvegarde sera perdu.
2. Lancez :

```
sudo /opt/read/code/deploy/restaurer.sh /var/backups/read/<horodatage> --confirmer
```

Le script arrête `app` et `caddy`, recrée la base, y restaure `base.dump`, remplace le contenu des volumes `app-donnees` et `app-keys`, puis relance tout par `docker compose up -d`. Le service migrations redonne au rôle de l'application ses droits et applique les migrations plus récentes que la sauvegarde.

3. Connectez-vous et contrôlez quelques fiches récentes.

6.4.3 Serveur perdu : rapatrier une copie distante

Sur un nouveau serveur installé comme au chapitre [Installation](#), avec le **même** fichier `.env` (mêmes variables `RCLONE_CONFIG_*` et même phrase de chiffrement) :

1. Déployez la plateforme (elle démarre sur une base vide).
2. Listez les copies (`lsd chiffre:`) et rapatriez la plus récente :

```
docker run --rm --env-file <(grep ^RCLONE_ /opt/read/.env) -v /var/backups/read:/r rclone/
rclone:1.75.1 \
  copy chiffre:<horodatage> /r/<horodatage>
```

3. Faites une restauration d'essai, puis la restauration réelle.
4. Pointez le nom de domaine vers le nouveau serveur.

CHAPITRE 7

Sécurité

Les mesures de sécurité en place dans la plateforme et sur le serveur, et ce qui reste de la responsabilité de l'exploitant.

Concerne : **Équipe technique** • Hébergeur

7.1 Transport et en-têtes

- **HTTPS obligatoire** : Caddy redirige HTTP vers HTTPS et renouvelle seul le certificat (Let's Encrypt). HTTP/2 et HTTP/3 sont servis.
- **En-têtes ajoutés par Caddy** : Strict-Transport-Security (un an, sous-domaines compris), X-Content-Type-Options: nosniff, X-Frame-Options: DENY, Referrer-Policy: strict-origin-when-cross-origin, Permissions-Policy (caméra, micro et géolocalisation refusés) ; l'en-tête Server est retiré.
- **Politique de sécurité du contenu (CSP)**, envoyée par l'application : scripts du site seulement, aucun script en ligne, aucune ressource ni connexion vers un autre site.

```
default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline'; img-src 'self' data: blob;; font-src 'self' data:; connect-src 'self'; object-src 'none'; base-uri 'self'; frame-ancestors 'self'; form-action 'self'
```

Le module hors ligne (/terrain/) a sa propre politique, qui ajoute 'wasm-unsafe-eval' (exigé par WebAssembly).

- **État de santé** : /health (qui interroge la base) est réservé au contrôle interne de Docker ; Caddy répond 404 à l'extérieur.

7.2 Comptes et sessions

- Mots de passe hachés (ASP.NET Identity), de 8 à 128 caractères dont un chiffre.
- **Verrouillage** : après 5 échecs, le compte est bloqué 15 minutes.
- **Limitation par adresse IP** : 10 tentatives de connexion par minute, 5 demandes d'inscription et 5 demandes « mot de passe oublié » par heure, 300 appels à l'API par minute (réglables, voir [Configuration](#)). Au-delà : réponse 429.
- **Session** : cookie read.session HttpOnly, Secure en HTTPS, SameSite=Lax, fermée après 30 minutes d'inactivité. Un compte désactivé ou dont le rôle change perd son accès dans la minute.
- **Lien « mot de passe oublié »** : valable une heure, utilisable une fois ; le message affiché est le même que le compte existe ou non.
- **Mot de passe temporaire** (compte créé par l'administrateur, premier administrateur) : à changer avant toute autre action.

- **API** : jetons seulement (jamais le cookie du site), compte revérifié à chaque appel (chapitre [API et module hors ligne](#)).

7.3 Autorisations et données

- Les autorisations (rôle et périmètre géographique) sont appliquées dans les services de l'application, jamais dans les seules pages : un écran caché n'est jamais la seule protection. Une donnée hors du périmètre du compte répond 404, une action non autorisée 403. Une matrice d'accès automatisée vérifie chaque route pour chaque rôle.
- **Base** : l'application se connecte avec un rôle restreint (`POSTGRES_APP_USER`) qui n'est pas propriétaire des tables et ne peut ni modifier ni effacer le journal d'audit. Le compte propriétaire ne sert qu'aux migrations et aux scripts de sauvegarde. Le port de la base n'est pas publié.
- **Journal d'audit** : chaque écriture, connexion, échec de connexion, export et téléchargement de fichier restreint y est inscrit, avec l'adresse IP réelle du client.
- **Protection des personnes** : aucune donnée nominative de bénéficiaire (effectifs agrégés) ; photos d'enfants sous consentement, sans métadonnées de localisation, jamais publiques ; listes de présence réservées à l'équipe projet et supprimées à échéance ; petits effectifs masqués sur le portail public.

7.4 Serveur

Mesures prises à l'[installation](#) :

- SSH par clé seulement, connexion `root` interdite ;
- pare-feu `ufw` : 22, 80 et 443 seulement ;
- mises à jour de sécurité automatiques du système ;
- secrets dans `/opt/read/.env` (droits 600), jamais dans le dépôt de code ni dans l'image ;
- conteneur de l'application exécuté sous un utilisateur non `root` ;
- journaux Docker bornés (5 fichiers de 10 Mo par conteneur).

7.5 Responsabilités de l'exploitant

Fréquence	Action
Chaque jour	Vérifier le compte rendu de la sauvegarde (copie hors serveur comprise)
Chaque mois	Mettre à jour les images (Mise à jour mensuelle) ; restauration d'essai
Chaque mois	Vérifier les comptes SSH autorisés (<code>~/ .ssh/authorized_keys</code>) et les comptes d'administrateur de la plateforme
À chaque départ	Retirer la clé SSH de la personne ; faire désactiver ses comptes par l'administrateur ; changer les secrets qu'elle connaissait
En cas d'incident	Conserver les journaux (Supervision) avant toute remise en état ; consulter le journal d'audit

Attention

Les clés de protection des données (volume `app-keys`) sont enregistrées en clair dans le volume, ce qui est acceptable sur un serveur dédié ; elles sont chiffrées dans la copie hors serveur. Ne copiez jamais ce volume ailleurs sans chiffrement.

7.6 Signaler ou traiter une compromission

Si vous soupçonnez un accès illégitime :

1. Gardez une copie des journaux (`docker compose logs`, volume `app-logs`) et faites une sauvegarde.
2. Faites désactiver par l'administrateur les comptes concernés ; changez les mots de passe exposés.
3. Pour couper tous les accès d'un coup (sessions et appareils du module hors ligne), changez les secrets de la base, puis remplacez les clés de protection des données : arrêtez `app`, videz le volume `app-keys`, redémarrez. Tous les utilisateurs devront se reconnecter.
4. Examinez le journal d'audit (écran **Journal d'audit** de l'administrateur) pour mesurer l'étendue des actions.

CHAPITRE 8

Supervision

Suivre l'état de la plateforme au quotidien : conteneurs, état de santé, journaux, place disque, sauvegardes et alertes conseillées.

Concerne : **Équipe technique** • Hébergeur

Les commandes de ce chapitre se lancent sur le serveur, dans le dossier du code :

```
cd /opt/read/code
```

8.1 État des conteneurs

```
docker compose ps -a
```

Exemple de résultat attendu :

NAME	SERVICE	STATUS
read-plateforme-app-1	app	Up 2 days (healthy)
read-plateforme-caddy-1	caddy	Up 2 days
read-plateforme-db-1	db	Up 2 days (healthy)
read-plateforme-migrations-1	migrations	Exited (0) 2 days ago

- `app` est contrôlé toutes les 15 secondes par Docker : il interroge `/health`, qui vérifie aussi l'accès à la base. Après 5 échecs, il passe « unhealthy ».
- `db` est contrôlé par `pg_isready`.
- `migrations` doit être arrêté avec le code 0 : un autre code signale un échec de migration ([Dépannage](#)).
- `app`, `db` et `caddy` redémarrent seuls après un arrêt inattendu ou un redémarrage du serveur (`restart: unless-stopped`).

8.2 Journaux

Source	Commande ou emplacement	Conservation
Application (console)	<code>docker compose logs --tail 100 app</code>	5 fichiers de 10 Mo
Application (fichiers quotidiens)	volume <code>read-plateforme_app-logs</code> , fichiers <code>read-AAAAMMJJ.log</code>	30 jours
Caddy (certificats, relais)	<code>docker compose logs --tail 100 caddy</code>	5 fichiers de 10 Mo
Base	<code>docker compose logs --tail 100 db</code>	5 fichiers de 10 Mo
Migrations	<code>docker compose logs migrations</code>	jusqu'au prochain déploiement

Déploiement	<code>/opt/read/déploiement.log</code>	dernier déploiement
Sauvegarde	<code>sudo journalctl -u read-sauvegarde.service</code>	journal systemd

Lire les fichiers quotidiens de l'application :

```
docker run --rm -v read-plateforme_app-logs:/logs:ro alpine:3.24 sh -c 'ls -l /logs && tail -n 50 /logs/$(ls /logs | tail -n 1)'
```

Chaque requête est journalisée sur une ligne (méthode, adresse, code de réponse, durée). Les niveaux `WRN` et `ERR` méritent attention, notamment :

- Échec de la tâche quotidienne : une tâche de conservation n'a pas abouti ;
- Aucun administrateur : renseigner `ADMIN_EMAIL` et `ADMIN_MOT_DE_PASSE` : base sans administrateur ;
- les erreurs d'envoi SMTP.

Remarque

Le contenu des e-mails n'est jamais journalisé (il peut porter un lien de réinitialisation de mot de passe) ; seul leur sujet l'est.

Le **journal d'audit** (actions des utilisateurs) se consulte dans la plateforme, écran **Journal d'audit** de l'administrateur ; il n'est pas dans les journaux techniques.

8.3 Place disque

```
df -h /
sudo du -sh /var/backups/read
docker system df
```

Les sauvegardes locales sont le premier poste de consommation (30 copies complètes de la base et des fichiers déposés). Les images Docker inutilisées laissées par les déploiements successifs s'accumulent aussi ; supprimez-les de temps en temps :

```
docker image prune -f
```

Danger

N'utilisez pas `docker system prune --volumes` ni `docker volume prune` : ces commandes peuvent supprimer les volumes de données de la plateforme.

8.4 Sauvegardes

Chaque jour ouvré, vérifiez que la sauvegarde de la nuit s'est terminée et que la copie hors serveur est partie :

```
sudo journalctl -u read-sauvegarde.service --since yesterday | tail -n 5  
ls /var/backups/read | tail -n 3
```

8.5 Alertes conseillées

La plateforme n'envoie pas elle-même d'alertes techniques. Mettez en place au minimum :

Surveillance	Moyen conseillé	Seuil
Disponibilité du site	Service de supervision externe interrogeant la page d'accueil en HTTPS toutes les 5 minutes	2 échecs consécutifs
Validité du certificat	Même service (contrôle du certificat)	Expiration dans moins de 14 jours (Caddy renouvelle bien avant)
Sauvegarde	Lecture quotidienne du compte rendu, ou script qui vérifie qu'un dossier de moins de 26 heures existe	Absence de sauvegarde du jour
Place disque	Supervision de l'hébergeur ou tâche planifiée sur d'f	Plus de 80 % occupés
Mémoire	Supervision de l'hébergeur	Usage du swap durable

8.6 Statistiques d'utilisation

L'administrateur dispose dans la plateforme d'un écran **Statistiques d'utilisation** (organisations actives, connexions par semaine, rapports transmis à temps, téléchargements). Il renseigne sur l'usage, pas sur la santé technique.

CHAPITRE 9

Dépannage

Fiches d'incidents courants, chacune avec ses symptômes, son diagnostic et sa solution.

Concerne : **Équipe technique** • **Hébergeur**

Chaque fiche décrit un **symptôme**, la façon de le **diagnostiquer** et la **solution**. Les commandes se lancent sur le serveur, dans `/opt/read/code`.

Conseil

Avant toute remise en état importante, faites une sauvegarde (`sudo /opt/read/code/deploy/sauvegarde.sh`) et gardez une copie des journaux : ils disparaissent quand les conteneurs sont recréés.

9.1 Le certificat HTTPS n'est pas obtenu

Symptôme. Le navigateur signale une connexion non sécurisée ou un certificat invalide ; le site ne répond pas en HTTPS.

Diagnostic.

```
docker compose logs --tail 200 caddy | grep -i -E "error|acme|challenge"
```

Causes habituelles :

- le nom de domaine ne pointe pas (ou pas encore) vers le serveur : vérifiez les enregistrements A et AAAA depuis un autre poste (`nslookup read.example.org`) ; un enregistrement AAAA vers une autre machine suffit à faire échouer la validation ;
- les ports 80 ou 443 sont fermés (pare-feu `ufw`, pare-feu de l'hébergeur) ou occupés par un autre service ;
- le domaine passe par un proxy ou un CDN ;
- `SITE_ADDRESS` contient une faute, un `https://` ou un `/` final ;
- trop d'essais : Let's Encrypt limite le nombre de demandes ; Caddy réessaie seul, plus tard.

Solution. Corrigez la cause, puis `docker compose up -d` (si `.env` a changé) ou `docker compose restart caddy`. Caddy redemande le certificat de lui-même.

9.2 Le conteneur de l'application est « unhealthy » ou redémarre en boucle

Symptôme. `docker compose ps` montre app « unhealthy » ou « Restarting » ; le site répond 502.

Diagnostic.

```
docker compose logs --tail 200 app
docker compose ps -a
```

- Chaîne de connexion 'ConnectionStrings:Read' manquante ou erreur d'authentification PostgreSQL pour read_app : le service migrations n'a pas créé ou mis à jour le rôle (voir la fiche suivante), ou POSTGRES_APP_PASSWORD a changé sans nouveau up.
- db n'est pas « healthy » : la base est arrêtée, le disque est plein, ou le volume est endommagé.
- Erreur au démarrage dans l'initialisation (référentiels, administrateur) : le message l'indique.

Solution. Corrigez la cause puis relancez l'ensemble : `docker compose up -d`. Si db est en cause, consultez `docker compose logs db`.

9.3 Les migrations échouent

Symptôme. migrations est arrêté avec un code différent de 0 ; app ne démarre pas (il attend la réussite des migrations).

Diagnostic.

```
docker compose logs migrations
```

Remarque

Sur une base vide, une erreur `SELECT ... FROM "__EFMigrationsHistory"` apparaît avant la création de cette table : elle est normale et n'arrête pas le service.

Causes possibles : POSTGRES_PASSWORD différent du mot de passe réel du compte propriétaire (il est fixé à la création du volume) ; nom de rôle POSTGRES_APP_USER invalide (lettres minuscules, chiffres et `_` seulement) ; migration en conflit avec des données existantes.

Solution. Corrigez `.env` puis `docker compose up -d`. Si une migration d'une nouvelle version échoue sur les données, revenez à la version précédente ([Retour arrière](#)) et restaurez au besoin la sauvegarde d'avant le déploiement ; transmettez le message d'erreur à l'équipe de développement.

9.4 Le disque est plein

Symptôme. Erreurs d'écriture, base arrêtée, dépôts de fichiers refusés, sauvegarde en échec.

Diagnostic.

```
df -h /
sudo du -sh /var/backups/read /var/lib/docker
docker system df
```

Solution.

1. Supprimez les images Docker inutilisées : `docker image prune -f`, et le cache de construction ancien : `docker builder prune -f --filter until=720h`.
2. Si les sauvegardes locales dominent et que la copie hors serveur est en place, réduisez

SAUVEGARDE_JOURS ou supprimez à la main les dossiers les plus anciens de `/var/backups/read`.

3. Si besoin, agrandissez le disque chez l'hébergeur.
4. Relancez `docker compose up -d` et vérifiez l'état des conteneurs.

Danger

Ne supprimez jamais de fichiers dans `/var/lib/docker/volumes` : ce sont les données de la plateforme.

9.5 Les e-mails ne partent pas

Symptôme. Les utilisateurs ne reçoivent ni le lien « mot de passe oublié » ni les copies de notifications.

Diagnostic.

```
docker compose logs --tail 300 app | grep -i -E "e-mail|smtp"
```

- écrit dans la boîte d'envoi locale ... aucun serveur SMTP réglé : EMAIL_HOTE est vide.
- Erreur d'authentification ou de connexion : identifiant, mot de passe, port ou EMAIL_SECURITE incorrects, ou port sortant bloqué par l'hébergeur.
- Messages envoyés mais non reçus : l'adresse EMAIL_EXPEDITEUR n'est pas autorisée par le serveur SMTP, ou les enregistrements SPF/DKIM du domaine manquent (messages classés indésirables).
- L'utilisateur n'a pas d'adresse e-mail dans son compte, ou a désactivé les copies par e-mail dans son profil.

Solution. Corrigez les variables EMAIL_*, puis `docker compose up -d`. Les notifications en attente sont envoyées par la tâche de fond qui tourne chaque minute.

9.6 L'heure affichée est fausse ou les connexions échouent par intermittence

Symptôme. Dates décalées dans le journal, jetons du module hors ligne refusés, certificat jugé « pas encore valide ».

Diagnostic. `timedatectl : System clock synchronized` doit valoir `yes`.

Solution. Réactivez la synchronisation : `sudo timedatectl set-ntp true`. La plateforme enregistre les dates en UTC et les affiche à l'heure de Niamey : ne changez pas le fuseau des conteneurs.

9.7 Le mot de passe de l'administrateur est perdu

Symptôme. Plus aucun administrateur ne peut se connecter.

Solution, dans l'ordre :

1. Un autre administrateur réinitialise le mot de passe depuis l'écran **Utilisateurs**.
2. Sinon, la personne utilise **Mot de passe oublié** sur la page de connexion ; le lien part à l'adresse e-mail du compte.

3. Sans serveur SMTP réglé, ce même e-mail est écrit dans le volume des fichiers déposés : affichez le plus récent pour y lire le lien (valable une heure) :

```
docker run --rm -v read-plateforme_app-donnees:/d:ro alpine:3.24 sh -c 'cat "$(ls -t /d/boite-envoi/*.eml | head -n 1)"'
```

Remarque

Après 5 échecs de connexion, un compte est bloqué 15 minutes : attendre suffit. La réinitialisation du mot de passe débloque aussi le compte.

ADMIN_EMAIL et ADMIN_MOT_DE_PASSE ne recréent un administrateur que s'il n'en existe **aucun** : ils ne servent pas à réinitialiser un compte existant.

9.8 Trop de requêtes (429) ou blocage d'un bureau entier

Symptôme. Des utilisateurs reçoivent la page « Trop de tentatives » à la connexion.

Diagnostic. Plusieurs personnes derrière la même adresse IP publique (bureau, partage de connexion) partagent la limite de 10 tentatives de connexion par minute.

Solution. Attendre une minute suffit. Si le cas est fréquent, augmentez `Securite__ConnexionsParMinute` ([Réglages avancés](#)).

9.9 Toutes les adresses du journal d'audit sont identiques

Symptôme. Le journal d'audit montre la même adresse IP pour tous les utilisateurs.

Diagnostic. Un relais placé devant Caddy (proxy ou CDN du gestionnaire DNS, répartiteur de charge de l'hébergeur) masque l'adresse des clients.

Solution. Passez l'enregistrement DNS en « DNS seulement ». Si un relais est indispensable, l'équipe de développement doit adapter Caddy et `Securite__ReseauxProxy` pour le reconnaître.

9.10 Une restauration d'essai échoue

Symptôme. `restaurer.sh --essai` affiche `ÉCHEC` ou une erreur de contrôle des empreintes.

Diagnostic.

- `sha256sum: WARNING ... did NOT match` : un fichier de la sauvegarde est altéré ou incomplet (copie interrompue).
- Différences de comptes : la sauvegarde ou sa restauration est incomplète.

Solution. Essayez la sauvegarde de la veille ou la copie hors serveur ; si plusieurs sauvegardes échouent, lancez une sauvegarde manuelle et observez son compte rendu (place disque, erreurs de `pg_dump`). Ne faites jamais une restauration réelle avec une sauvegarde dont l'essai a échoué.

9.11 Les utilisateurs voient une ancienne version du module hors ligne

Symptôme. Après un déploiement, le module hors ligne d'un téléphone n'a pas les nouveautés.

Solution. Le module garde son application en cache pour fonctionner sans réseau. À la connexion suivante avec du réseau, il détecte la nouvelle version et propose **Mettre à jour** ; elle s'active aussi quand tous les onglets du module sont fermés. Voir [Mise à jour du module](#).

CHAPITRE 10

API et module hors ligne

L'API REST sécurisée par jetons, ses durées et ses limites, et le fonctionnement du module hors ligne côté exploitation.

Concerne : **Équipe technique**

10.1 L'API REST

La plateforme expose une API sous `/api/v1`. Elle sert d'abord au module hors ligne, et peut servir à d'autres outils de l'organisation. Elle passe par les mêmes services que le site, donc par les **mêmes règles** : rôle du compte, périmètre géographique, paramètre « saisie par les organisations », montants masqués selon le rôle.

Réponse	Signification
401	Jeton absent, expiré ou révoqué (compte désactivé, droits ou mot de passe changés, appareil déconnecté)
403	Action non autorisée pour ce compte ; mot de passe temporaire à changer d'abord sur le site
404	Donnée hors du périmètre du compte, ou inexistante
409	Donnée modifiée entre-temps par quelqu'un d'autre
429	Trop d'appels depuis la même adresse IP

Les erreurs sont au format `ProblemDetails` (JSON) ; les énumérations sont écrites en texte (par exemple "Realisee").

10.1.1 Points d'accès

Route	Méthodes	Usage
<code>/api/v1/jetons</code>	POST	Connexion : jeton d'accès et jeton de renouvellement
<code>/api/v1/jetons/renouveler</code>	POST	Nouveau couple de jetons ; le jeton présenté ne sert plus
<code>/api/v1/jetons/revoquer</code>	POST	Déconnexion de l'appareil
<code>/api/v1/moi</code>	GET	Profil du compte connecté
<code>/api/v1/referentiels</code>	GET	Listes de saisie
<code>/api/v1/organisations</code> , <code>/api/v1/organisations/{id}</code>	GET	Organisations visibles
<code>/api/v1/activites</code>	GET, POST	Liste filtrée et paginée ; planification
<code>/api/v1/activites/a-rapporter</code>	GET	Activités pour lesquelles le compte peut saisir un rapport

/api/v1/activites/{id}	GET, PUT	Fiche, modification
/api/v1/activites/{id}/statut	PUT	Avancement
/api/v1/rapports	GET, POST	Liste ; saisie (brouillon, soumission ou validation)
/api/v1/rapports/{id}	GET, PUT	Fiche, correction
/api/v1/rapports/{id}/renvoi	POST	Renvoi d'un rapport à l'organisation (gestion)

La description complète (OpenAPI) est servie sur /api/v1/openapi.json, aux comptes connectés seulement (par le site ou par jeton).

10.1.2 Jetons

1. Le client obtient ses jetons par POST /api/v1/jetons avec l'identifiant et le mot de passe d'un compte :

```
{ "identifiant": "personne@example.org", "motDePasse": "...", "appareil": "Téléphone de terrain 3" }
```

2. Il reçoit un **jeton d'accès** (jetonAcces, valable expireDansSecondes) à présenter dans l'en-tête Authorization: Bearer <jeton>, et un **jeton de renouvellement** (jetonRenouvellement, valable jusqu'à renouvellementExpireLe).
3. Avant l'expiration du jeton d'accès, il échange le jeton de renouvellement contre un nouveau couple (/api/v1/jetons/renouveler).

Règle	Valeur par défaut	Réglage
Durée du jeton d'accès	30 minutes	Api__DureeJetonAccesMinutes
Durée du jeton de renouvellement, glissante	14 jours	Api__DureeRenouvellementJours
Durée maximale avant de redemander le mot de passe	90 jours	Api__DureeMaximaleJours
Rejeu toléré d'un jeton de renouvellement déjà utilisé	2 minutes	Api__DelaiRejeuSecondes
Appels par minute et par adresse IP	300	Api__AppelsParMinute
Connexions et renouvellements par minute et par adresse IP	10	Securite__ConnexionsParMinute

Sécurité des jetons :

- la zone /api n'accepte que les jetons, jamais le cookie du site ;
- le jeton de renouvellement est à **usage unique** et seule son empreinte est enregistrée en base ;
- un jeton de renouvellement présenté une seconde fois au-delà du délai de rejeu est considéré comme volé : tous les jetons de l'appareil sont révoqués ;
- le compte est révérifié à chaque appel : désactivation, changement de rôle ou de mot de passe coupent l'accès aussitôt ;
- les connexions par l'API sont inscrites au journal d'audit (canal « api ») et comptent dans les statistiques.

Les jetons expirés ou révoqués depuis plus de 30 jours sont supprimés par la tâche quotidienne.

Remarque

Il n'existe pas d'écran listant les appareils connectés. Pour couper l'accès d'un téléphone perdu, l'administrateur désactive le compte (ou change son mot de passe) : tous ses jetons sont refusés à l'appel suivant.

10.2 Le module hors ligne

Le module (`/terrain/`) est une application WebAssembly installable (PWA), servie par la plateforme elle-même. Il est réservé aux comptes qui saisissent : l'équipe projet, et les organisations quand l'administrateur a ouvert la saisie aux organisations.

10.2.1 Ce qui est stocké sur le téléphone

- l'application elle-même, en cache (service worker) : seules les adresses de `/terrain/` passent par ce cache, l'API va toujours au réseau ;
- dans le stockage du navigateur (IndexedDB) : le profil, les listes de saisie, les activités à rapporter (au plus 500 : sans rapport, non annulées, commençant dans les 30 jours ou terminées depuis 90 jours au plus), les brouillons de rapport et le jeton de renouvellement. Le jeton d'accès reste en mémoire seulement.

Les rapports validés sont retirés du téléphone 30 jours après leur validation. **Se déconnecter de ce téléphone** révoque les jetons sur la plateforme puis les efface de l'appareil.

10.2.2 Synchronisation

L'envoi n'a lieu que **module ouvert** : au démarrage, au retour du réseau, toutes les 5 minutes tant que des rapports attendent, et sur **Synchroniser maintenant**. Un rapport envoyé deux fois (réponse perdue sur un réseau instable) n'est jamais dupliqué : la plateforme reconnaît son identifiant et renvoie le rapport existant.

10.2.3 Mise à jour du module

Un déploiement de la plateforme publie aussi la nouvelle version du module. Sur chaque téléphone :

1. à la connexion suivante avec du réseau, le service worker télécharge la nouvelle version ;
2. le module affiche « Une nouvelle version du module est prête. » et le bouton **Mettre à jour** (les saisies en cours sont enregistrées d'abord) ;
3. sans action de l'utilisateur, la nouvelle version s'active quand tous les onglets du module sont fermés.

Les fichiers du module sont servis avec `Cache-Control: no-cache` : le navigateur vérifie toujours la version disponible.

Important

Un déploiement qui change le format des données de l'API doit rester compatible avec l'ancienne version du module tant que des téléphones ne sont pas mis à jour : des rapports peuvent attendre plusieurs jours dans leur file.

CHAPITRE 11

Environnement de démonstration

Installer une plateforme de démonstration avec des données fictives, la remettre à zéro, et ce qu'il ne faut jamais faire sur une plateforme réelle.

Concerne : **Équipe technique**

Une plateforme de démonstration s'installe exactement comme une plateforme réelle ([Installation](#)), avec trois réglages en plus. Elle sert aux présentations et aux formations.

11.1 Données de démonstration

Avec `DEMO_DONNEES=true` et un `DEMO_MOT_DE_PASSE`, l'application charge au démarrage un jeu de données **entièrement fictif** : organisations, écoles, subventions, micro-projets, environ 300 activités sur l'année en cours, environ 200 rapports, évaluations, indicateurs, messages, bonnes pratiques et ressources. Les statuts (activités en retard, rapports en attente) sont calculés par rapport au jour du chargement.

Le chargement n'a lieu **que si la base ne contient encore aucune organisation**. Le journal de l'application le signale par un avertissement `Données de démonstration FICTIVES chargées`.

Comptes de démonstration (mot de passe `DEMO_MOT_DE_PASSE`, sauf l'administrateur) :

Compte	Rôle
ADMIN_EMAIL (par exemple admin@demo.read)	Administrateur initial, mot de passe ADMIN_MOT_DE_PASSE (créé comme sur une plateforme réelle)
gestion@demo.read	Équipe projet (gestion)
odr@demo.read	Organisation
comite@demo.read	Comité communal
partenaire@demo.read	Partenaire institutionnel

Réglages propres à la démonstration dans `/opt/read/.env` :

Variable	Valeur
DEMO_DONNEES	true
DEMO_MOT_DE_PASSE	8 caractères au moins, dont un chiffre
DEMO_AFFICHER	true : bandeau « Données de démonstration fictives » sur toutes les pages
DEMO_COMPTES_PROTEGES	true : un visiteur ne peut ni changer le mot de passe d'un compte ...@demo.read (profil, mot de passe oublié, administrateur), ni le modifier, ni le désactiver
ADMIN_CHANGEMENT_OBLIGATOIRE	false : les comptes partagés ne doivent pas exiger un changement de mot de passe

Conseil

Lisez le mot de passe de démonstration sur le serveur sans l'afficher à l'écran d'une présentation, par exemple dans une variable : `MDP=$(grep '^DEMO_MOT_DE_PASSE=' /opt/read/.env | cut -d= -f2-)`.

11.2 Remettre la démonstration à zéro

Après une présentation ou des essais, remettez les données dans leur état de départ :

```
/opt/read/code/deploy/reinitialiser-demo.sh --confirmer
```

Le script arrête l'application, recrée une base vide, efface les fichiers déposés et redémarre : le schéma, les référentiels, les comptes et les données fictives sont rechargés (une minute environ). Il refuse de s'exécuter si `DEMO_DONNEES` ne vaut pas `true`.

Remarque

Après une remise à zéro, les dates sont recalculées par rapport au jour même : les retards et les échéances restent réalistes.

11.3 Ce qu'il ne faut jamais faire sur une plateforme réelle

Danger

- Ne réglez jamais `DEMO_DONNEES=true` sur une plateforme réelle : sur une base encore vide, des données fictives s'y mêleraient aux vraies.
- Ne copiez jamais le fichier `.env` d'une démonstration vers un serveur réel (mots de passe partagés, bandeau, changement de mot de passe non exigé).
- Ne lancez jamais `reinitialiser-demo.sh` en dehors d'une démonstration : il efface toutes les données.
- Ne restaurez jamais une sauvegarde de démonstration sur une plateforme réelle, ni l'inverse.

Une démonstration et une plateforme réelle doivent tourner sur des serveurs distincts (le projet Compose a toujours le même nom, `read-plateforme`).

ANNEXE A

Aide-mémoire des commandes

Les commandes d'exploitation les plus utiles, regroupées par tâche.

Concerne : **Équipe technique**

Sauf mention contraire, les commandes se lancent sur le serveur, dans `/opt/read/code`.

A.1 État et journaux

```
cat /opt/read/code/VERSION           # commit déployé
docker compose ps -a                 # état des conteneurs
docker compose logs --tail 100 app    # journal de l'application
docker compose logs --tail 100 caddy  # certificats, relais
docker compose logs migrations       # dernières migrations
tail -f /opt/read/deploiement.log     # déploiement en cours
```

A.2 Démarrer, arrêter, appliquer un réglage

```
docker compose up -d                 # démarrer ou appliquer un changement de .env
docker compose restart app            # redémarrer l'application seule
docker compose stop                   # arrêter (les données sont conservées)
```

Danger

Jamais `docker compose down -v` : l'option `-v` supprime les volumes, donc toutes les données.

A.3 Déploiement (depuis le poste)

```
deploy/deployer.sh <alias-ssh>
SSH=/c/Windows/System32/OpenSSH/ssh.exe deploy/deployer.sh <alias-ssh> # sous Windows (Git Bash)
```

Retour à la version précédente (sur le serveur) :

```
cd /opt/read && rm -rf code && mv code.ancien code && cd code && docker compose up -d --build
```

A.4 Sauvegarde et restauration

```
sudo /opt/read/code/deploy/sauvegarde.sh # sauvegarde
```

```

immédiate
systemctl list-timers read-sauvegarde.timer                                # prochaine
sauvegarde
sudo journalctl -u read-sauvegarde.service -n 20                        # dernier compte
rendu
sudo ls /var/backups/read                                                # sauvegardes locales
sudo /opt/read/code/deploy/restaurer.sh /var/backups/read/<horodatage> --essai    # essai,
sans effet
sudo /opt/read/code/deploy/restaurer.sh /var/backups/read/<horodatage> --confirmer  # REMPLACE
les données
docker run --rm --env-file <(grep ^RCLONE_ /opt/read/.env) rclone/rclone:1.75.1 lsd chiffre:  #
copies distantes

```

A.5 Base de données

```

docker compose exec db psql -U read -d read                                # console SQL
(compte propriétaire)
docker compose exec db psql -U read -d read -c "SELECT count(*) FROM organisations"

```

Attention

La console SQL donne tous les droits, y compris sur le journal d'audit. Ne modifiez jamais les données par SQL : toute écriture doit passer par la plateforme pour être contrôlée et tracée.

A.6 Fichiers et e-mails locaux

```

docker run --rm -v read-plateforme_app-logs:/logs:ro alpine:3.24 ls -l /logs    # journaux
quotidiens
docker run --rm -v read-plateforme_app-donnees:/d:ro alpine:3.24 ls -lt /d/boite-envoi  # e-
mails non envoyés (sans SMTP)

```

A.7 Système et place disque

```

df -h / && sudo du -sh /var/backups/read                                # place disque
docker system df                                                        # place utilisée par
Docker
docker image prune -f                                                  # images inutilisées
timedatectl                                                            # horloge
synchronisée
sudo ufw status                                                         # pare-feu

```

A.8 Démonstration seulement

```

/opt/read/code/deploy/reinitialiser-demo.sh --confirmer                # remise à zéro
(refusée hors démo)

```