Dokos
Applications libres de gestion d'entreprise

Dossier de conception générale

Dokos

Dokos SAS
7B Allée neuve 54520 Laxou
SIREN - 824710776

Votre contact
Charles-Henri Decultot
06.33.47.58.69
chdecultot@dokos.io

Versions du document

Version	Date	Objet	Responsable	Approbateur
v1.0	01.10.2025	Version initiale du document	Charles-Henri Decultot	Charles-Henri Decultot
v1.0.1	22.01.2026	Remplacement de requirements.txt par pyproject.toml + précisions système de version. MAJ du num de version	Charles-Henri Decultot	Charles-Henri Decultot
v1.1	07.08.2026	Passage de l'application Certification Caisse en version majeure 6.0.0 (NC 26-4) ; divergence assumée avec la numérotation du cadriciel et déclaration de compatibilité ; précisions sur l'empreinte de version; mise à jour du lexique.	Charles-Henri Decultot	Charles-Henri Decultot

Version du logiciel

Nom	Version
Dodock (Frappe)	5.3.0
Dokos	5.3.0
Certification Caisse	6.0.0

Classification du document

Public sous licence CC-by-nc-sa

SOMMAIRE

Introduction **5**
1. Présentation générale **5**
1.1. Vue d'ensemble de Dokos 5
1.2. Historique et fondations techniques 5
1.3 Couverture fonctionnelle 6
1.4 Utilisateurs de Dokos 7
1.5 Spécificités Françaises 8
1.6 Objectif du document 8
1.7 Public cible 8
1.8 Contexte réglementaire 9
2. Architecture technique du logiciel 9
2.1 Vue d'ensemble de l'architecture 9
Technologies 10
Infrastructure 10
Architecture générale 11
Architecture de déploiement web 13
Architecture applicative 14
Concepts principaux 16
DocType (Type de document) 16
Gestion des rôles et autorisations 16
2.2 Organisation du code 17
Structure racine d'un site Dodock 17
Structure d'une application Dodock 17
Organisation des DocTypes 18
Structure d'un site 19
Dossiers spécialisés 19
2.3 Matériel d'encaissement 20
3. Périmètre fonctionnel du logiciel 20
4. Périmètre de la certification 23
4.1 Utilisateurs concernés par le périmètre de certification 23
4.2 Application et modules inclus dans le périmètre de certification 24
Architecture de la certification 24
Cadriciel Dodock 25

L'application Certification Caisse..... 25
Modules fonctionnels Dokos concernés..... 26
A. Module de ventes..... 26
B. Module de comptabilité..... 26
Flux de données certifié..... 26
Périmètre d'utilisation certifié..... 27
4.3 Versions du logiciel..... 27
Système de versions du logiciel..... 27
Version concernée par la présente certification..... 28
Suivi et objectifs des versions..... 29
Identification de la version..... 29
Empreinte de version..... 30
Annexes..... 31
Annexe 1 - Lexique..... 31
Annexe 2 - Principaux Modèles Logiques de Données..... 37

Introduction

Ce document de conception générale a pour objectif de décrire l'architecture fonctionnelle et technique du logiciel Dokos dans le cadre de sa certification par le Laboratoire National de métrologie et d'Essais (LNE) pour les logiciels d'encaissement. Il détaille les spécifications et les choix techniques retenus pour garantir la conformité de Dokos aux exigences légales et réglementaires en vigueur, notamment en matière d'inaltérabilité, de sécurisation, de conservation et d'archivage des données d'encaissement.[1]

1. Présentation générale

1.1. Vue d'ensemble de Dokos

Dokos est un logiciel de gestion d'entreprise (ERP) open source, conçu pour répondre aux besoins des petites, moyennes et grandes entreprises. Il constitue une adaptation française d'ERPNext, un ERP reconnu mondialement, bénéficiant de la robustesse et de la flexibilité du cadriciel Frappe sur lequel il est développé.
Cette architecture modulaire permet à Dokos d'offrir une solution complète et évolutive, capable de couvrir un large éventail de besoins métiers tout en respectant les spécificités réglementaires françaises.

1.2. Historique et fondations techniques

ERPNext et son cadriciel applicatif Frappe sont développés depuis 2008 par l'équipe de Frappe Technologies, basée à Mumbai (Inde). Cette solution a acquis une réputation internationale grâce à sa couverture fonctionnelle et sa flexibilité.

[1] Exigences 8, 10, 12, 16 et 17 du référentiel LNE

Depuis 2019, la société Dokos SAS maintient et développe une version dérivée de Frappe/ERPNext, spécifiquement adaptée aux exigences du marché français. Cette démarche permet d'intégrer les modifications nécessaires pour assurer la conformité avec la réglementation française, tout en préservant les avantages de la solution originale.

Dans le présent document, nous distinguerons la version française maintenue par Dokos SAS de la version originale avec les termes suivants:

- **Dodock** : désigne la version adaptée du cadriciel Frappe
- **Dokos** : désigne la version adaptée de l'application ERPNext

À certains endroits nous pourrons faire référence à Dokos plutôt qu'à Dodock, car l'application Dokos s'appuie entièrement sur le cadriciel Dodock pour son fonctionnement.

1.3 Couverture fonctionnelle

Dokos propose une suite complète de modules couvrant l'ensemble des processus d'entreprise.

Modules commerciaux

- Gestion des ventes et devis
- Gestion de la relation client (CRM)
- E-commerce simple
- Support client
- Portail client

Modules de gestion

- Achats et approvisionnements
- Gestion des stocks et entrepôts
- Sous-traitance
- Gestion de projets
- Gestion de la qualité
- Portail fournisseur

Modules financiers et comptables

- Comptabilité générale, analytique et auxiliaire
- Gestion des immobilisations
- Contrôle budgétaire
- Reporting financier

Modules de production

- Planification et ordonnancement
- Gestion de production manufacturière
- Nomenclatures et gammes

Modules transversaux

- Gestion électronique des documents
- Reporting et analyse de données
- Workflows d'approbation transactionnels
- Traçabilité des opérations
- API REST
- Oauth / OpenID Connect pour les intégrations et connexions SSO

1.4 Utilisateurs de Dokos

Dokos est un logiciel libre, conçu pour être accessible et adaptable à tout type d'organisation, quelle que soit sa taille. Sa structure modulaire permet à chaque entité d'activer uniquement les fonctionnalités dont elle a besoin, ce qui explique la diversité des usages constatés.

Les principales catégories d'utilisateurs de Dokos sont :

- **Les très petites entreprises (TPE)** : souvent composées de quelques personnes, elles recherchent un outil simple et économique pour gérer leur facturation, leur comptabilité et parfois leur relation client, sans avoir à multiplier les logiciels.

- **Les petites et moyennes entreprises (PME)** : disposant de processus plus structurés, elles utilisent Dokos pour centraliser leurs opérations (ventes, achats, stocks, ressources

humaines) et gagner en efficacité dans leur gestion quotidienne.

- **Les grandes entreprises** : certaines l'emploient comme un cadre applicatif (cadriciel) pour développer ou personnaliser leurs propres solutions internes, tandis que d'autres s'appuient sur des modules spécifiques tels que la gestion des achats pour renforcer leurs processus.

- **Les associations** : qu'elles soient assujetties à la TVA ou non, elles trouvent dans Dokos un moyen de gérer leurs adhésions, leurs financements, leur comptabilité et parfois leurs activités commerciales, tout en bénéficiant de la souplesse du logiciel libre.

1.5 Spécificités Françaises

La version Dokos intègre des fonctionnalités spécifiquement développées pour le marché français, incluant notamment la conformité comptable, la gestion de la TVA française et les obligations réglementaires relatives aux logiciels de caisse.

1.6 Objectif du document

L'objectif de ce dossier est de fournir une vue d'ensemble fonctionnelle, métier et technique de la solution Dokos[2], en exposant :

- L'architecture générale technique de la plateforme Dokos
- L'architecture générale applicative et modulaire
- Les modules concernés par les opérations de vente, d'encaissement et de facturation
- Le périmètre fiscal et les versions concernées par la certification LNE
- Le mécanisme de *versioning* appliqué au logiciel Dokos et ses applications

1.7 Public cible

Ce dossier de conception général s'adresse principalement aux :

[2] Exigence 1

- Auditeurs ou organismes certificateurs souhaitant analyser les mécanismes internes de Dokos dans le cadre de certifications
- Équipes de développement et contributeurs du projet open source Dokos
- Intégrateurs et partenaires techniques accompagnant les déploiements Dokos
- Utilisateurs professionnels (directions informatiques, responsables comptables, etc.) cherchant à comprendre le fonctionnement détaillé de Dokos pour une intégration dans un environnement contrôlé ou réglementé
- Organismes de contrôle fiscal dans le cadre d'audits ou de vérifications de conformité

1.8 Contexte réglementaire

Dokos, en tant que solution ERP intégrant des fonctionnalités de caisse et de facturation, doit répondre aux exigences réglementaires françaises, notamment :

- Article 88 de la loi de finances 2016 relatif aux logiciels de caisse[3]
- Obligation d'inaltérabilité des données de caisse
- Traçabilité des opérations commerciales et comptables
- Archivage sécurisé des données fiscales
- Conformité TVA et déclarations fiscales automatisées

Ce dossier détaille comment Dokos implémente ces obligations réglementaires.

2. Architecture technique du logiciel

2.1 Vue d'ensemble de l'architecture

Dodock est un cadriciel web moderne conçu pour le développement rapide d'applications métiers. Dokos est construit sur ce cadriciel et hérite de toutes ses capacités techniques.

Dodock et Dokos s'appuient sur une architecture web de type client-serveur. Ils sont accessibles directement depuis un navigateur et offrent une grande adaptabilité, garantissant

[3] Loi n° 2015-1785 du 29 décembre 2015 de finances pour 2016 – Article 88

leur compatibilité avec différents environnements matériels et systèmes d'exploitation pour l'utilisateur.

Voici un récapitulatif des technologies utilisées pour son fonctionnement et de l'infrastructure nécessaire pour installer et déployer le logiciel.

Technologies

Serveur (Backend)

- **Language**: Python 3.11+
- **Framework web** : Cadriciel Dodock (fondé sur Werkzeug)
- **Base de données** : MySQL/MariaDB (support PostgreSQL en développement)[4]
- **Serveur web** : Nginx (proxy inverse)
- **Serveur d'application** : Gunicorn WSGI
- **Cache** : Redis (sessions, cache applicatif, *queues*)
- **Système de file d'attente** : Redis Queue (RQ) pour les tâches asynchrones
- **Système d'exploitation** : Linux (Debian v11+ / Ubuntu 22.04+)

Client (Frontend)

- **Language** : JavaScript / Vue.js 3 (API de composition)
- **CSS** : Bootstrap 4/5 personnalisé
- **Templates** : Jinja2 pour les pages générées côté serveur

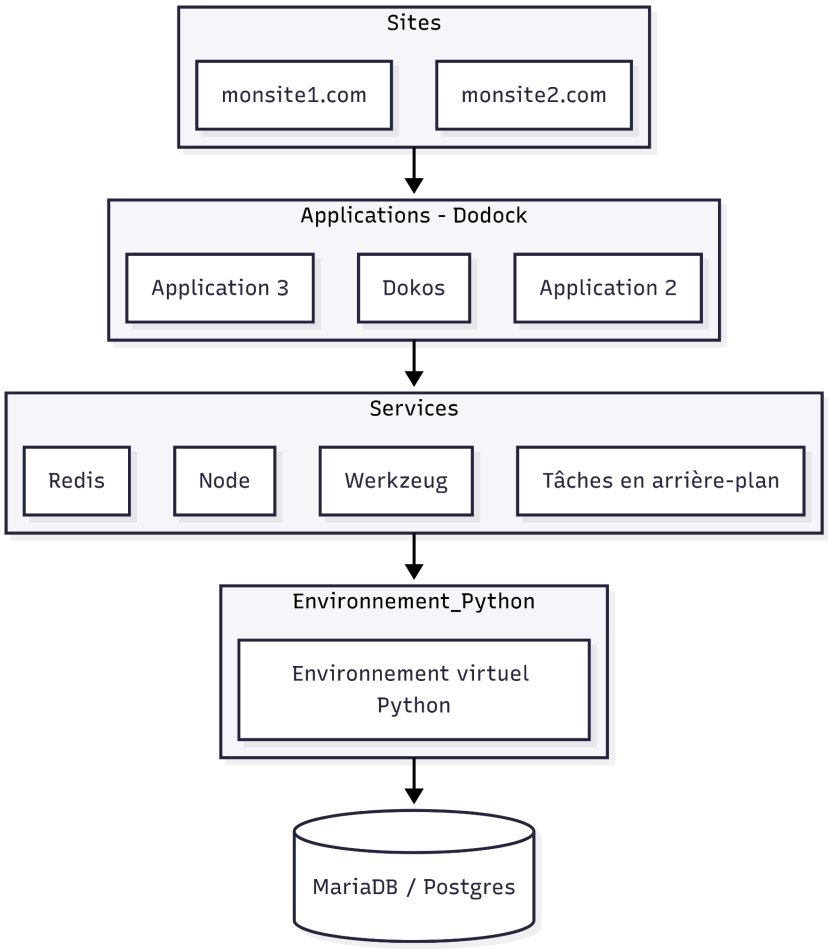
Le logiciel peut être utilisé depuis n'importe quel poste client via un navigateur web compatible (Firefox, Chrome, etc.). L'accès s'effectue par une connexion sécurisée (HTTPS), assurant la protection et l'intégrité des données.

Infrastructure

- **Conteneurisation** : Docker
- **Supervision** : Intégration Prometheus/Grafana/Elasticsearch
- **Journaux techniques** : Logging structuré Python
- **Sécurité** : HTTPS, protection CSRF par jeton, *rate limiting*

[4] Exigence 16

Architecture générale



Ce diagramme illustre l'architecture multi-tenant de Dodock/Dokos, organisée en couches hiérarchiques.

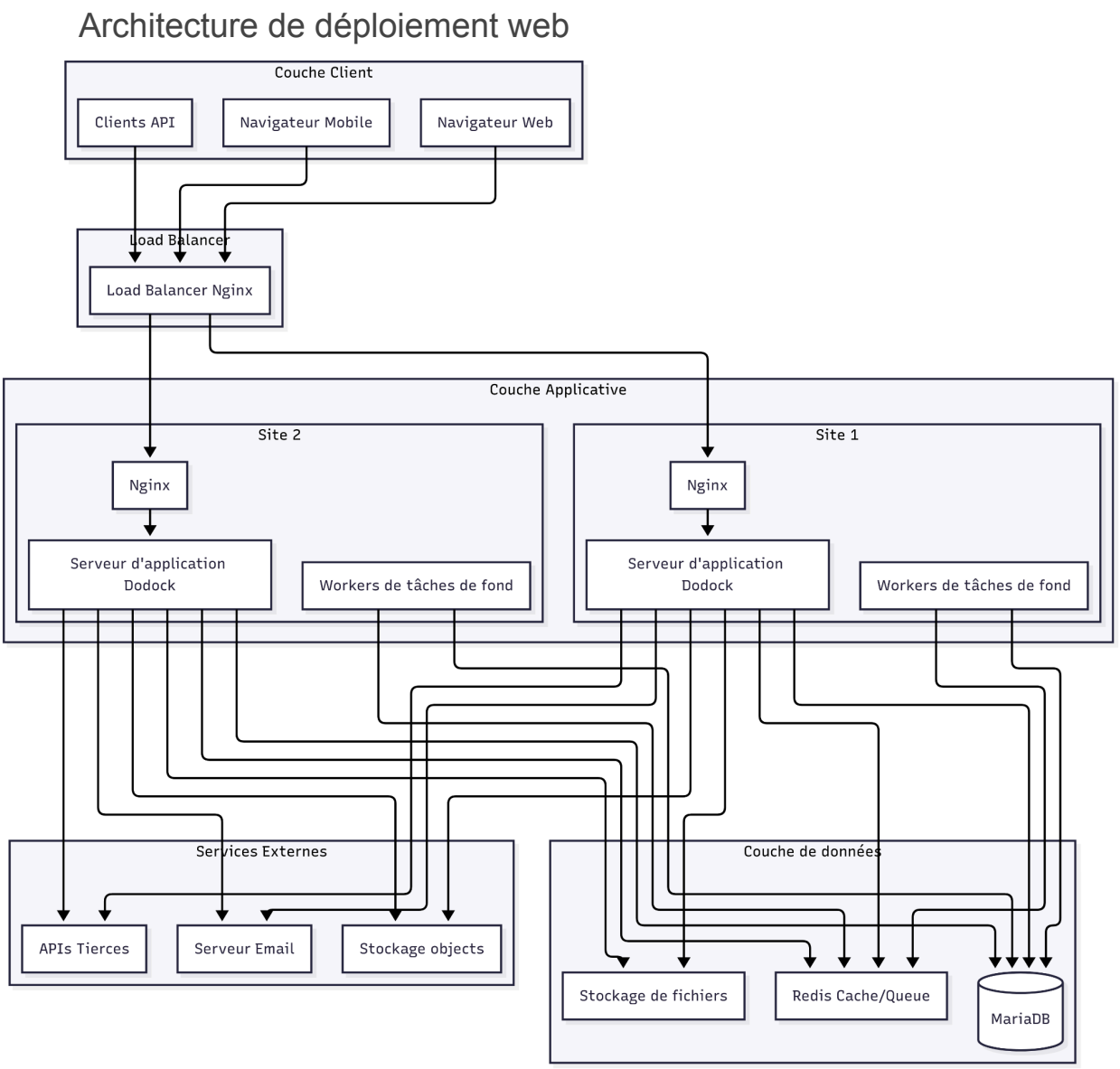
Au sommet, la couche **Sites** permet d'héberger plusieurs instances distinctes (monsite1.com, monsite2.com) sur la même infrastructure, chacune avec ses propres données et configuration. Ces sites partagent la couche **Applications** qui contient Dokos et d'autres applications métiers installées. Chaque application est elle-même composée d'un ou plusieurs modules.

La couche **Services** fournit les services techniques essentiels : Node.js pour les outils de développement ou le build des éléments javascript, Werkzeug comme serveur WSGI Python,

Redis pour le cache et les sessions, ainsi que le système de tâches en arrière-plan pour les traitements asynchrones.

L'ensemble s'exécute dans un **Environnement Python** isolé via un environnement virtuel, garantissant la cohérence des dépendances. Enfin, tous les sites partagent l'accès à la base de données **MariaDB/PostgreSQL** qui stocke les données de manière cloisonnée par site.

Cette architecture permet une gestion efficace de multiples entreprises sur une même installation tout en maintenant l'isolation des données et la mutualisation des ressources techniques.



Ce diagramme présente l'architecture technique complète d'un déploiement Dokos en production, structurée en couches distinctes pour assurer haute disponibilité et scalabilité.

La **Couche Client** regroupe tous les points d'accès : navigateurs web et mobiles pour les utilisateurs finaux, ainsi que les clients API pour les intégrations tierces. Ces requêtes transitent

par un **Load Balancer Nginx** qui distribue intelligemment la charge entre les instances disponibles.

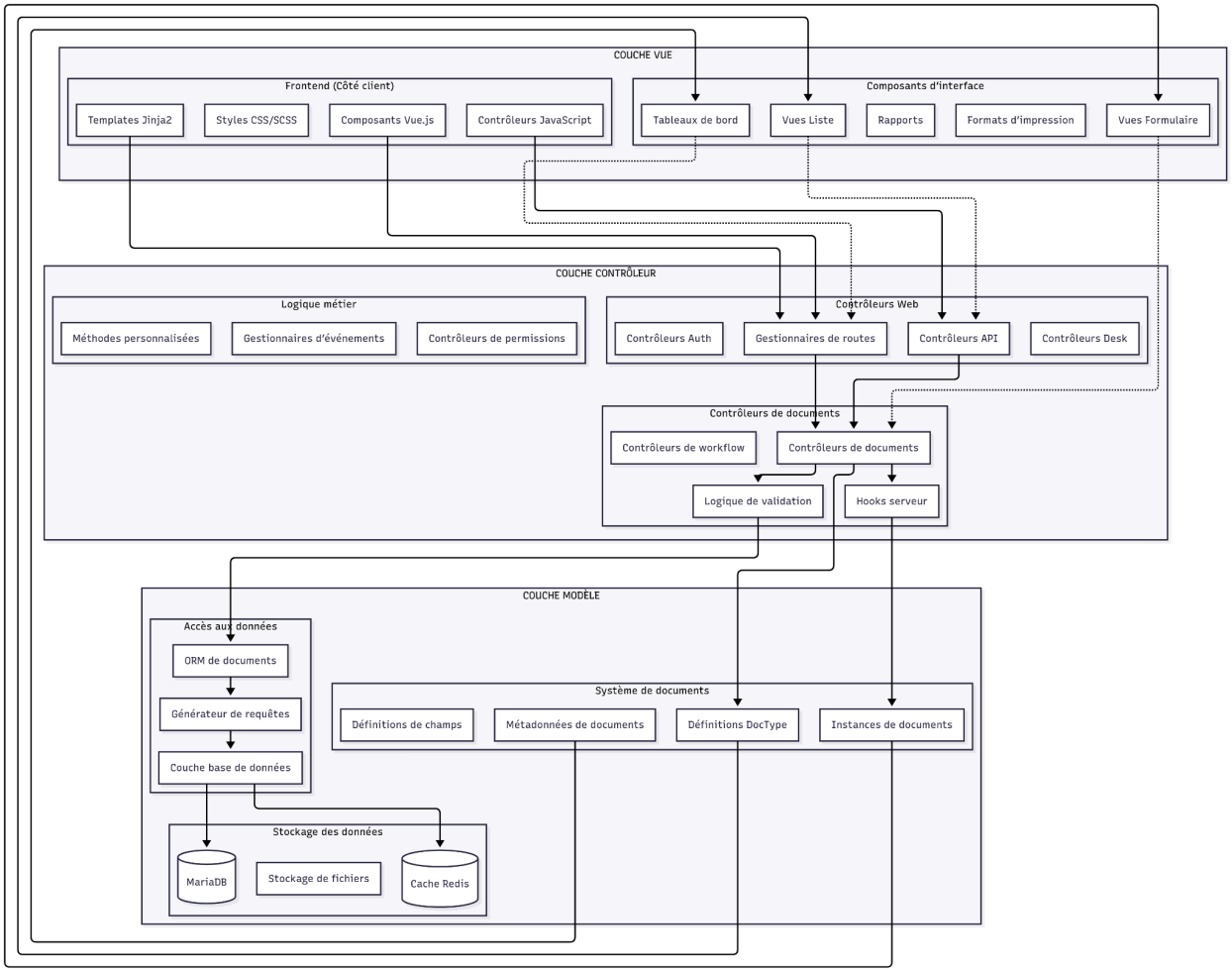
La **Couche Applicative** comprend plusieurs sites Dokos déployés en parallèle (Site 1 et Site 2), chacun disposant de son propre serveur web Nginx, de son serveur d'application Dokos, et de workers dédiés aux tâches de fond. Cette architecture permet une montée en charge horizontale et une résilience en cas de défaillance d'un site.

La **Couche de Données** centralise le stockage avec Redis pour le cache et les files d'attente, MariaDB pour les données métier, et un système de stockage de fichiers. Cette centralisation permet aux différents sites de partager les ressources tout en maintenant la cohérence des données.

Enfin, les **Services Externes** incluent le serveur email pour les notifications, le stockage d'objets pour les fichiers volumineux, et les APIs tierces pour les intégrations métier. Cette architecture modulaire garantit performance, disponibilité et facilité de maintenance pour les déploiements Dokos d'entreprise.

Architecture applicative

Dodock implémente une architecture MVC (Model-View-Controller) modernisée et adaptée aux applications web métier. Cette architecture sépare clairement les responsabilités entre la gestion des données, l'interface utilisateur et la logique métier.



Ce diagramme illustre l'architecture en couches de l'application web organisée selon le modèle **MVC (Modèle – Vue – Contrôleur)**.

La **couche Vue** regroupe le frontend côté client, construit avec des composants Vue.js, des templates Jinja2, des contrôleurs JavaScript et des styles CSS/SCSS. Elle fournit les différents éléments d'interface utilisateur comme les vues formulaires, les listes, les tableaux de bord, les rapports et les formats d'impression.

La **couche Contrôleur** orchestre la logique applicative. On y distingue les contrôleurs web (gestionnaires de routes, API, authentification, interface Bureau), les contrôleurs de documents

(gestion des documents, hooks serveur, logique de validation et workflow) ainsi que la logique métier (gestionnaires d'événements, permissions et méthodes personnalisées).

Enfin, la **couche Modèle** gère la structure et la persistance des données à travers le système de documents (définitions de DocType, instances de documents, métadonnées, champs), l'accès aux données (ORM, générateur de requêtes, couche de base de données) et le stockage (MariaDB, Redis pour le cache et un espace de fichiers).

Les composants de la vue appellent les contrôleurs, qui eux-mêmes interagissent avec le modèle pour lire, valider ou modifier les données, avant de restituer l'information à l'interface utilisateur.

Concepts principaux

DocType (Type de document)

La particularité du cadriciel Dodock est de proposer des métadonnées dynamiques, grâce au concept de DocType, stockées directement en base de données comme des documents ordinaires.

La création d'un DocType permet de définir la structure d'un document (formulaire) et également de générer automatiquement différents types de vues (Liste, Rapport, Gantt, Calendrier, etc…) ainsi qu'une API Rest dédiée.

Gestion des rôles et autorisations

Le système de permissions de Dodock est granulaire et contextuel, permettant un contrôle fin des accès.
Il est possible de définir des autorisations via un système de rôles et de profil de rôles avec des restrictions pouvant aller jusqu'aux champs de formulaire.

Des restrictions peuvent également être appliquées sur certaines valeurs pour limiter le périmètre d'accès des utilisateurs. *Exemple: Accès aux documents d'une seule société.*

2.2 Organisation du code

Le cadriciel Dodock utilise une structure de fichiers et dossiers bien organisée qui suit des conventions spécifiques. Voici comment s'organise un projet Dodock typique :

Structure racine d'un site Dodock

```
None
dodock-bench/
├── apps/                    # Applications installées
├── sites/                   # Sites individuels
├── config/                  # Configuration globale
├── logs/                    # Fichiers de log
└── env/                     # Environnement virtuel Python
```

Structure d'une application Dodock

Chaque application dans le dossier `apps/` suit cette organisation :

```
None
mon_app/
├── mon_app/
│   ├── __init__.py
│   ├── hooks.py             # Configuration des hooks
│   ├── modules.txt          # Liste des modules
│   ├── patches.txt          # Scripts de migration
│   ├── config/
│   │   ├── desktop.py       # Configuration du bureau
│   │   └── docs.py          # Configuration de la documentation
│   ├── public/              # Fichiers statiques accessibles
publiquement
│   │   ├── css/
```

```
│   │   ├── js/
│   │   └── images/
│   ├── templates/          # Templates Jinja2
│   │   ├── emails/         # Templates d'emails
│   │   ├── pages/          # Templates de pages web
│   │   └── includes/       # Templates réutilisables
│   ├── www/                # Pages web statiques
│   └── modules/            # Modules métier
│       └── mon_module/
│           ├── __init__.py
│           ├── doctype/    # Types de documents
│           ├── page/       # Pages personnalisées
│           ├── report/     # Rapports
│           ├── web_form/   # Formulaires web
│           └── dashboard_chart_source/
├── pyproject.toml          # Dépendances Python
├── setup.py                # Configuration du package
└── license.txt
```

Organisation des DocTypes

Chaque DocType a sa propre structure dans `modules/mon_module/doctype/` :

```
None
mon_doctype/
├── __init__.py
├── mon_doctype.py          # Logique métier (classe Python)
├── mon_doctype.js          # Scripts côté client
├── mon_doctype.json        # Définition du formulaire
├── mon_doctype_list.js     # Personnalisation de la vue liste
├── mon_doctype_calendar.js # Vue calendrier
```

```
├── mon_doctype_tree.js    # Vue arborescence
└── test_mon_doctype.py    # Tests unitaires
```

Structure d'un site

Dans le dossier `sites/`, chaque site a sa propre organisation :

```
None
mon_site/
├── site_config.json       # Configuration du site
├── private/
│   ├── files/             # Fichiers privés
│   └── backups/           # Sauvegardes
├── public/
│   └── files/             # Fichiers publics téléversés
├── error.log              # Logs d'erreurs
├── web.log                # Logs web
└── locks/                 # Fichiers de verrouillage
```

Dossiers spécialisés

Templates : Organisés par type d'utilisation

- `templates/emails/` : Templates pour les emails automatiques
- `templates/pages/` : Templates pour les pages web
- `www/` : Pages statiques accessibles via URL

Assets statiques :

- `public/css/` : Feuilles de style personnalisées
- `public/js/` : Scripts JavaScript personnalisés
- `public/images/` : Images et autres médias

Configuration :

- `hooks.py` : Point d'entrée pour personnaliser le comportement
- `modules.txt` : Liste des modules de l'application
- `patches.txt` : Scripts de migration de données

Cette structure modulaire permet une organisation claire du code, facilite la maintenance et respecte les principes de séparation des préoccupations propres au cadriciel Dodock.

2.3 Matériel d'encaissement[5]

Dokos n'est pas lié à un type de matériel spécifique et son utilisation ne requiert généralement aucun équipement supplémentaire pour encaisser.
Néanmoins, il peut être associé à différents périphériques — liste non exhaustive — tels que :

- **une imprimante à tickets** pour éditer reçus, avoirs ou factures,
- **un lecteur de codes-barres** pour scanner les produits ou les cartes de fidélité,
- **une imprimante à étiquette** pour identifier et coder les produits simplement.

Il n'existe pas de connecteur natif avec des terminaux de paiement bancaires ni avec les tiroirs-caisses automatiques.

3. Périmètre fonctionnel du logiciel

Dokos est une plateforme logicielle complète, combinant les fonctionnalités d'un **CRM (Customer Relationship Management - Outil de gestion de la relation client)** et d'un **ERP (Enterprise Resource Planning - Outil de planification des ressources de l'entreprise)**.
Elle se positionne comme une solution intégrée pour gérer et optimiser l'ensemble des processus d'une entreprise, de la prospection client à la comptabilité, en passant par la gestion des opérations et des ressources.

[5] Exigence 1

Voici une description détaillée du périmètre fonctionnel de Dokos :

CRM (Gestion de la Relation Client)

- **Gestion des Contacts et des Prospects :**
  - Stockage centralisé des informations clients (coordonnées, historique, préférences, etc.).
  - Segmentation et classification des contacts.
  - Importation et exportation de données.
  - Gestion des pistes (Lead) et opportunités de vente (flux de vente visuel, suivi des étapes).
  - Attribution des pistes et opportunités aux équipes commerciales.
  - Suivi des interactions clients (appels, e-mails, réunions).
- **Gestion des Ventes :**
  - Création et gestion des devis et des commandes.
  - Suivi des négociations et des propositions commerciales.
  - Automatisation du processus de vente (workflows, déclencheurs).
  - Gestion des tarifs et des remises.
  - Reporting et analyses des ventes (tableaux de bord, indicateurs clés de performance).
- **Gestion du Support Client :**
  - Gestion des tickets de support (création, attribution, suivi, résolution).
  - Base de connaissances pour les clients et les agents de support.
  - Gestion des SLA (Service Level Agreements - Garanties de niveau de service).
  - Intégration avec différents canaux de communication (e-mail, chat, téléphone).
- **Marketing :**
  - Gestion des campagnes marketing (e-mailing).
  - Segmentation des clients pour cibler les campagnes.

ERP (Gestion des Ressources de l'Entreprise) :

- **Gestion des Stocks :**
  - Suivi des stocks en temps réel.
  - Gestion des approvisionnements (commandes fournisseurs, réceptions).
  - Gestion des inventaires (inventaires périodiques, valorisation des stocks).
  - Alertes de réapprovisionnement.

- **Gestion des Achats :**
  - Gestion des fournisseurs (informations, tarifs, conditions).
  - Création et gestion des bons de commande.
  - Suivi des réceptions et des factures fournisseurs.
  - Analyse des dépenses et optimisation des coûts.
- **Gestion de la Production (si applicable) :**
  - Gestion des nomenclatures (BOM - Bill of Materials).
  - Planification de la production (MRP - Material Requirements Planning).
  - Suivi de l'avancement de la production.
- **Gestion Financière :**
  - Comptabilité générale (journal, grand livre, balance).
  - Gestion des factures (création, envoi, suivi des paiements).
  - Gestion des encaissements et des décaissements.
  - Rapports financiers (bilan, compte de résultat, trésorerie).
  - Budget et prévisions financières.
- **Gestion des Ressources Humaines (RH) :**
  - Gestion des employés (informations personnelles, contrats, salaires).
  - Gestion des congés et des absences.
  - Gestion des évaluations et des performances.
- **Gestion de Projets :**
  - Création et gestion de projets.
  - Suivi des tâches et des délais.
  - Allocation des ressources aux projets.
  - Gestion des coûts des projets.
  - Suivi de l'avancement des projets.

Intégrations et Personnalisation :

- **Intégrations :**
  - Intégration avec des outils tiers (e-commerce, paiements en ligne, outils collaboratifs, etc.).
  - API (Application Programming Interface) pour les intégrations personnalisées.
- **Personnalisation :**
  - Personnalisation des champs et des formulaires.
  - Création de rapports et de tableaux de bord personnalisés.
  - Configuration des workflows et des automatisations.
  - Adaptation de l'interface utilisateur (UI) à la marque de l'entreprise.

Autres Fonctions Essentielles :

- **Sécurité :** Gestion des accès et des rôles utilisateurs, authentification sécurisée.
- **Mobilité :** Accès aux données et aux fonctionnalités via des vues adaptées avec ou sans PWA (progressive web application).
- **Reporting et Analyse :** Tableaux de bord personnalisables, rapports prédéfinis et personnalisés.
- **Collaboration :** Fonctionnalités de collaboration intégrées (chat, commentaires, partage de documents).
- **Gestion Documentaire :** Stockage et gestion des documents (GED).
- **Import/Export de données :** Fonctionnalités d'import et d'export des données du logiciel sous format standard (CSV/XLSX)

4. Périmètre de la certification

4.1 Utilisateurs concernés par le périmètre de certification[6]

Dokos est un logiciel de facturation doté d'une fonctionnalité de saisie des paiements. L'enregistrement d'un paiement déclenche concomitamment, automatiquement, obligatoirement, instantanément et sans intervention humaine, l'enregistrement d'une écriture comptable dans le système de comptabilité.
Cependant, nombre d'utilisateurs de Dokos n'utilisent pas Dokos pour saisir leur comptabilité légale, mais uniquement comme système de facturation. Les factures et leur information de paiement sont ensuite envoyées à leur comptable ou expert comptable pour enregistrement final dans le système de comptabilité légale de l'entreprise.
Cependant, techniquement, ces écritures déclenchent toutes concomitamment, automatiquement, obligatoirement, instantanément et sans intervention humaine, l'enregistrement d'une écriture comptable dans le système de comptabilité de Dokos. Il n'est pas possible de désactiver cette mécanique d'enregistrement.

La certification LNE vise à garantir à cette typologie d'utilisateurs, que l'utilisation de Dokos dans ce cadre est bien en conformité avec la législation française.

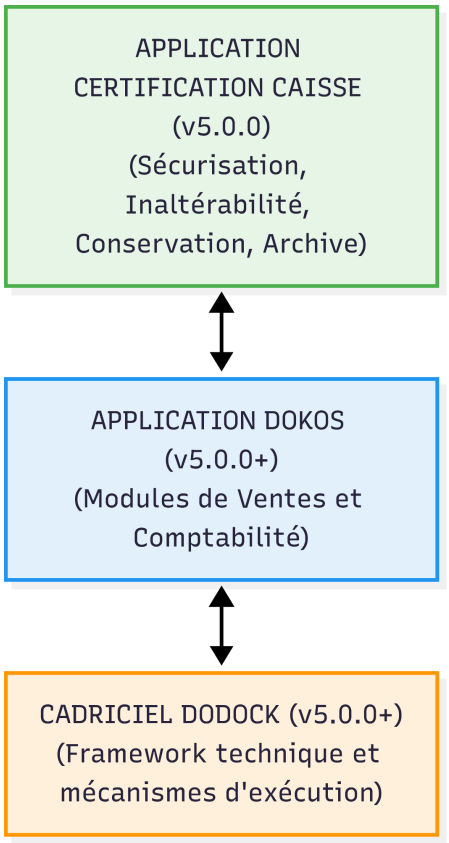
[6] Définition de la fonctionnalité de caisse selon le référentiel LNE

4.2 Application et modules inclus dans le périmètre de certification[7]

La couverture fonctionnelle de Dokos dépasse le cadre légal appliqué aux logiciels d'encaissement. De ce fait, le périmètre de certification sera limité aux modules et éléments suivants, qui constituent le cœur de la conformité réglementaire.

Architecture de la certification

Le système certifié repose sur trois couches complémentaires :



[7] Exigences 20 et 21

Cadriciel Dodock

Le cadriciel Dodock constitue la fondation technique du système certifié.
Il fournit :

- Les mécanismes d'exécution et de persistance des données
- Le système de gestion des DocTypes et des métadonnées
- Les hooks d'événements permettant d'intercepter les opérations
- L'infrastructure de base de données (MariaDB/PostgreSQL)
- Le système de gestion des droits et autorisations

Le cadriciel Dodock est certifié en version 5.3.0 et versions mineures suivantes (5.x.x), garantissant la stabilité du socle technique.

L'application Certification Caisse

L'application Certification Caisse est le composant central de la conformité réglementaire. Elle s'appuie sur les mécanismes d'exécution du cadriciel Dodock pour garantir les quatre caractéristiques exigées par la réglementation :

1. **Inaltérabilité**[8] : Chaînage cryptographique des données d'encaissement
2. **Sécurisation** : Protection des données contre toute modification ou suppression
3. **Conservation**[9] : Maintien des données pendant la durée légale requise
4. **Archivage**[10] : Export des données dans un format ouvert et sécurisé

Cette application est déployée spécifiquement pour les clients entrant dans le périmètre de certification. Elle est certifiée en version 6.0.0.

Fonctionnement : L'application Certification Caisse utilise les hooks du cadriciel Dodock pour intercepter automatiquement tout enregistrement, modification ou suppression de données d'encaissement, et applique les mécanismes de sécurisation appropriés de manière transparente pour l'utilisateur.

[8] Exigence 8
[9] Exigence 16
[10] Exigences 10, 11 et 12

Modules fonctionnels Dokos concernés

Bien que l'application Dokos offre de nombreux modules (CRM, achats, stocks, production, etc.), seuls deux modules sont concernés par le périmètre de certification car ils génèrent des données d'encaissement :

A. Module de ventes[11]

Ce module gère le processus commercial complet :

- Base de clients
- Base articles (Produits/Services)
- Devis et commandes
- Factures de vente [12]
- Enregistrements de paiements
- Vue Point de Vente (interface d'encaissement)

B. Module de comptabilité[13]

Ce module enregistre les écritures comptables générées par les encaissements :

- Écritures comptables (General Ledger Entry)
- Grand livre comptable
- Rapports financiers

Note importante : La certification ne couvre que les fonctionnalités liées à l'enregistrement extra-comptable des paiements et leur traçabilité. Les autres fonctionnalités comptables (gestion budgétaire, analytique, déclarations fiscales) ne sont pas concernées par le périmètre de certification.

Il est à noter, que le logiciel certifié dans son ensemble est **Dokos v5**, le cadriciel et l'application Certification Caisse étant les couches techniques permettant de garantir le respect des exigences fiscales.

Flux de données certifié

L'ordre suivant illustre le flux des données d'encaissement dans le système certifié :

1. L'utilisateur crée une facture dans le Module Ventes (Dokos)

[11] Exigence 3
[12] Exigence 9
[13] Exigence 3

2. L'utilisateur enregistre un paiement
3. L'application Certification Caisse intercepte automatiquement ces événements
4. Les données sont sécurisées (hash, signature, chaînage)
5. Une écriture comptable est générée dans le Module Comptabilité
6. Les données sécurisées sont conservées dans la base de données
7. À la demande, l'application Certification Caisse génère des archives conformes

Périmètre d'utilisation certifié

Le périmètre de certification cible spécifiquement les utilisateurs qui sont concernés par la loi :
- Utilisent Dokos comme logiciel de facturation uniquement
- N'utilisent PAS Dokos pour saisir leur comptabilité légale
- Transmettent leurs factures et paiements à leur expert-comptable

Pour ces utilisateurs, la certification LNE garantit que l'utilisation de Dokos pour la facturation et l'encaissement est conforme à l'article 88 de la loi de finances 2016, même si la comptabilité légale est tenue dans un autre système.

Cas d'usage typique : Une TPE/PME utilise Dokos pour gérer ses ventes et encaissements au quotidien, puis transmet mensuellement un export de ses factures et paiements à son cabinet comptable qui les saisit dans le logiciel de comptabilité légale de l'entreprise.

4.3 Versions du logiciel

Système de versions du logiciel

Dokos suit un système de numérotation de version selon les règles de la gestion sémantique (https://semver.org/lang/fr/).

Étant donné un numéro de version MAJEUR.MINEUR.CORRECTIF, les modifications de numérotations représentent le type de changement suivant :

- le numéro de version MAJEUR quand il y a des changements non rétrocompatibles

- le numéro de version MINEUR quand il y a des ajouts de fonctionnalités rétrocompatibles
- le numéro de version de CORRECTIF quand il y a des corrections d'anomalies rétrocompatibles

La présente certification n'entraîne aucune modification de cette règle pour le cadriciel Dodock.

En revanche, pour l'application **Certification Caisse**, la règle Semver est complétée par une contrainte réglementaire prioritaire : toute modification du périmètre fiscal certifié entraîne obligatoirement l'incrémentation du numéro de version majeure, l'information préalable du LNE et une évaluation complémentaire, conformément aux exigences 21 du référentiel et aux modalités de surveillance du certificat (cf. Dossier 5, §6.3-6.4).

L'application Certification Caisse a été publiée, pour sa première version, avec un numéro de version identique à celui du cadriciel, afin d'éviter toute ambiguïté de version.

Cet alignement ne peut être maintenu au-delà : la contrainte réglementaire énoncée ci-dessus impose à Certification Caisse une version majeure à chaque modification du périmètre fiscal, indépendamment du rythme de publication du cadriciel. Les deux numérotations divergent donc à partir de la première modification du périmètre fiscal. Chaque version de Certification Caisse déclare en conséquence la version de Dodock et de Dokos avec laquelle elle est compatible, et c'est cette déclaration, et non l'égalité des numéros, qui identifie sans ambiguïté la combinaison certifiée.

Version concernée par la présente certification

La version du logiciel concernée est la suivante:

Cadriciel Dodock[14]
V5.3.0 ainsi que les versions mineures suivantes.
Application Dokos[15]
V5.3.0 ainsi que les versions mineures suivantes.

[14] Exigence 21
[15] Exigence 21

Application Certification Caisse[16]
V6.0.0

Seule l'utilisation conjointe du cadriciel Dodock, de l'application Dokos et de l'application Certification Caisse permet d'identifier le logiciel certifié. La combinaison certifiée est : Dodock V5.3.0 et versions mineures suivantes, Dokos V5.3.0 et versions mineures suivantes, et Certification Caisse V6.0.0.

Suivi et objectifs des versions

Nouvelle version MAJEURE : toute modification de l'application affectant le périmètre de certification entraîne obligatoirement la publication d'une version majeure du logiciel.

Nouvelle version MINEURE : elle assure qu'aucune modification n'impacte le périmètre fiscal et que toutes les exigences réglementaires sont respectées.

Nouvelle version CORRECTIVE : elle garantit également l'absence d'impact sur le périmètre fiscal et le respect des exigences applicables.

Identification de la version

Les numéros de versions des applications concernées par le périmètre fiscal sont disponibles dans les fichiers __init__.py[17] dans le dossier de l'application:

- Dodock : https://gitlab.com/dokos/dodock/-/blob/develop/frappe/__init__.py
- Certification Caisse : https://gitlab.com/dokos/certification_caisse/-/blob/develop/certification_caisse/__init__.py

Ils sont également accessibles aux utilisateurs ayant le rôle "Responsable Système" depuis le menu d'**Aide > A Propos**, disponible dans l'interface utilisateur de l'application.

[16] Exigence 21
[17] Exigence 21

Empreinte de version[18]

À chaque publication officielle d'une nouvelle version de Dodock ou de Certification Caisse, un fichier est généré et publié dans le répertoire de l'application correspondante. Ce fichier contient une empreinte cryptographique des fichiers relevant du périmètre fiscal.

Il comprend notamment :

- la liste complète des fichiers inclus dans le périmètre fiscal et de certification
- l'empreinte cryptographique associée à chaque fichier

Ce fichier est régénéré à chaque publication, et son contenu change dès lors que le contenu d'un fichier du périmètre fiscal a été modifié — même lorsque la liste des fichiers, elle, est inchangée. Une évolution qui ajoute ou retire un fichier du périmètre modifie à la fois la liste et les empreintes ; une modification d'un mécanisme existant ne modifie que les empreintes. Les deux cas sont donc distinguables par simple comparaison de deux fichiers d'empreinte successifs. La procédure de régénération, de vérification et de déclaration est décrite au Dossier 5, §6.6.

Ce fichier constitue une **preuve d'intégrité** des fichiers contenus dans la version de l'application et est disponible dans le dossier source de l'application :

- https://gitlab.com/dokos/dodock
- https://gitlab.com/dokos/certification_caisse

La signature systématique des fichiers publiés a pour objectif de **prévenir toute fraude** via modification du code source. Elle permet également aux auditeurs et contrôleurs de **vérifier rapidement l'intégrité** des fichiers inclus dans le périmètre de certification.

Un **outil dédié** est mis à disposition des auditeurs et contrôleurs pour effectuer cette vérification depuis le serveur, via la ligne de commande. Les modalités d'utilisation de cet outil sont détaillées dans le dossier utilisateur.

[18] Exigence 21

Annexes

Annexe 1 - Lexique

Terme	Définition
Document	Objet contenant des données et enregistré dans la base de données du logiciel.
Type de document	Objet contenant des données et enregistré dans la base de données du logiciel. Le terme est parfois interchangé avec DocType ou Type de document dans Dodock/Dokos
DocType	Objet contenant des données et enregistré dans la base de données du logiciel. Le terme est parfois interchangé avec DocType ou Type de document dans Dodock/Dokos
Application	Logiciel conçu pour accomplisser une ou plusieurs tâches spécifiques Ici nous parlons uniquement d'application conçues et déployées spécifiquement sur le framework/cadriciel Dodock
Module	Sous-ensemble d'une application déployées sur le cadriciel Dodock Ex. Module de ventes
Framework	Terme courant pour désigner un cadriciel
Cadriciel	Un cadriciel (ou framework en anglais) est un ensemble d'outils et de composants logiciels organisés conformément à un plan d'architecture et des patterns, l'ensemble formant ou promouvant un « squelette » de programme. Dodock est le cadriciel sur lequel sont développées toutes les applications citées dans ce dossier.
ORM	ORM (Object-Relational Mapping) désigne une technique de programmation informatique qui crée un pont entre les bases de données relationnelles et le paradigme de la programmation orientée objet. L'ORM permet aux développeurs d'interagir avec une base de données à l'aide d'objets, sans avoir à écrire directement des

	instructions SQL ou à gérer les échanges de données de manière manuelle.
Anomalie Bloquante	Défaillance du service rendant impossible son utilisation ou empêchant une fonctionnalité essentielle de fonctionner, sans solution de contournement
Anomalie Majeure	Dysfonctionnement affectant significativement le fonctionnement du service sans le rendre totalement inutilisable, mais nécessitant une intervention rapide.
Anomalie Mineure	Dysfonctionnement ou imperfection n'affectant pas de manière significative le fonctionnement global du service ou la disponibilité des fonctionnalités principales.
Approbation croisée	Processus de validation par lequel deux personnes distinctes examinent et approuvent une modification avant sa mise en production.
Archivage	Conservation à long terme des données et des documents dans un format sécurisé et accessible pour répondre aux obligations légales et réglementaires. Elle porte à la fois sur les données d'encaissement, exportables sous forme d'archive fiscale signée, et sur les fichiers PDF originaux des justificatifs émis, conservés sous un nom dérivé de leur empreinte et vérifiés à chaque relecture.
Bitwarden	Gestionnaire de mots de passe open-source utilisé obligatoirement par l'équipe DOKOS SAS.
CSIRT (Computer Security Incident Response Team)	Équipe de réponse aux incidents de sécurité informatique.
Dodock	Cadriciel (framework) sur lequel est basé le logiciel DOKOS.
DPO (Data Protection Officer)	Délégué à la Protection des Données, responsable de la conformité RGPD.
ERP (Enterprise Resource Planning)	Progiciel de gestion intégré permettant de gérer l'ensemble des processus opérationnels d'une entreprise.
ERPNext	Logiciel ERP open-source sur lequel est basé DOKOS.
Frappe	Écosystème technologique sur lequel reposent ERPNext et DOKOS.
Gameplan	Outil de gestion de projet interne à DOKOS SAS.
GitLab	Plateforme de gestion de versions et de collaboration pour le développement logiciel.

Hetzner	Fournisseur d'hébergement allemand utilisé par DOKOS SAS.
Helpdesk	Système de gestion des tickets et incidents interne à Dokos
Logs	Journaux d'événements enregistrant les actions et opérations effectuées sur les systèmes.
Merge request	Demande de fusion de code dans un système de gestion de versions
On-premise	Mode de déploiement où le logiciel est installé sur les serveurs du client.
Open-source	Modèle de développement logiciel où le code source est accessible, modifiable et redistribuable.
OVH	Fournisseur d'hébergement français utilisé par DOKOS SAS.
PCA (Plan de Continuité d'Activité)	Document définissant les procédures permettant d'assurer la continuité des services en cas d'incident majeur.
Post-mortem	Analyse rétrospective d'un incident permettant d'en comprendre les causes et d'identifier des améliorations.
Press	Logiciel de provisionnement automatique des serveurs utilisé par DOKOS Cloud.
Prometheus/Grafana	Outils de supervision et de monitoring des infrastructures.
Raven	Outil de communication interne à DOKOS SAS.
RGPD (Règlement Général sur la Protection des Données)	Réglementation européenne encadrant le traitement des données personnelles.
Rollback	Procédure de retour à une version antérieure du système en cas de problème.
RPO (Recovery Point Objective)	Perte de données maximale acceptable en cas d'incident.
RTO (Recovery Time Objective)	Durée maximale d'interruption acceptable d'un service.
SaaS (Software as a Service)	Mode de distribution où le logiciel est hébergé par l'éditeur et accessible via internet.
Sécurisation	Ensemble des mesures techniques et organisationnelles visant à protéger les données contre les accès non autorisés.
SSO (Single Sign-On)	Système d'authentification unique permettant d'accéder à plusieurs applications avec un seul identifiant.

Tiers-lieu	Espace de travail partagé (coworking, fablab, etc.) constituant une clientèle spécifique de DOKOS.
UFW (Uncomplicated Firewall)	Pare-feu déployé sur les serveurs DOKOS Cloud.
Versionnement sémantique	Système de numérotation des versions au format MAJEUR.MINEUR.CORRECTIF. Pour l'application Certification Caisse, cette règle est complétée par une contrainte réglementaire prioritaire : toute modification du périmètre fiscal impose une version majeure, indépendamment de l'ampleur de la modification, et s'accompagne de l'information préalable du LNE, d'une évaluation complémentaire et de la republication de l'empreinte de version. La numérotation majeure de Certification Caisse diverge de ce fait de celle du cadriciel Dodock ; la compatibilité entre les deux est déclarée par chaque version.
Wiki	Base de connaissances collaborative interne à l'organisation.
Bench	Outil de ligne de commande développé par Frappe permettant de gérer les environnements de développement et les instances Dokos (installation, mise à jour, gestion des sites).
CERT-FR	Centre gouvernemental de veille, d'alerte et de réponse aux attaques informatiques, rattaché à l'ANSSI.
Code-owner	Système de notification automatique désignant un responsable spécifique pour valider les modifications touchant un périmètre défini du code source.
Conservation	Maintien des données d'encaissement pendant les durées légales requises avec garantie d'intégrité et d'accessibilité.
Docker	Technologie de conteneurisation permettant d'encapsuler des applications et leurs dépendances dans des environnements isolés et reproductibles.
ESLINT	Outil d'analyse statique du code JavaScript permettant d'identifier et corriger les problèmes de qualité de code.
Fork	Version dérivée d'un logiciel open-source, maintenue de manière indépendante tout en conservant un lien avec le projet source.
Frappe Technologies	Entreprise indienne développant le framework Frappe et le logiciel ERPNext.
Issue	Ticket dans GitLab documentant un bug, une demande d'évolution ou une tâche de développement.

Loi n° 2015-1785 (Article 88)	Loi de finances pour 2016 imposant l'utilisation de logiciels de caisse certifiés pour les assujettis à la TVA.
Milestone	Jalon de projet dans GitLab regroupant des issues et définissant une version ou une étape majeure du développement.
NF 525	Norme française définissant les caractéristiques techniques des systèmes de caisse certifiés (inaltérabilité, sécurisation, conservation, archivage).
NTP	Network Time Protocole, protocole réseau permettant d'obtenir une heure fiable depuis un système externe tel qu'une horloge atomique.
Nuage	Plateforme propriétaire développée par Dokos SAS pour automatiser le déploiement et la maintenance des instances Dokos hébergées.
PDP Libre (Plateforme de Dématérialisation Partenaire)	Association regroupant des acteurs open-source pour l'implémentation de la facturation électronique obligatoire.
PEP8	Convention de style pour le code Python, définissant les règles de formatage et d'organisation du code
PRETTIER	Outil de formatage automatique du code JavaScript garantissant une cohérence stylistique.
RACI (Responsible, Accountable, Consulted, Informed)	Matrice définissant les rôles et responsabilités pour chaque activité d'un processus.
REX (Retour d'EXpérience)	Analyse formalisée d'un projet ou incident permettant de capitaliser les enseignements et d'améliorer les pratiques.
RUFF	Outil moderne de contrôle et formatage du code Python, combinant rapidité et exhaustivité des vérifications.
SEMGREP	Outil d'analyse statique avancée du code permettant de détecter des patterns de sécurité et de qualité.
SMC (Système de Management de la Conformité)	Ensemble documenté de processus, procédures et responsabilités garantissant le respect des exigences réglementaires.
Sprint	Période de développement de durée fixe durant laquelle l'équipe réalise un ensemble défini de fonctionnalités.
Ticket	Enregistrement structuré d'une demande de support, d'un bug ou d'une tâche dans un système de suivi.

Traçabilité	Capacité à reconstituer l'historique complet des opérations, modifications et décisions grâce à des enregistrements horodatés et nominatifs. Pour les justificatifs, elle porte sur les émissions : chaque remise, originale ou duplicata, est enregistrée avec son horodatage, son utilisateur, son canal et l'empreinte du document remis.
Veille réglementaire	Processus de surveillance continue des évolutions légales et réglementaires impactant l'activité de Dokos SAS.
Versioning	Gestion des différentes versions du code source avec historisation complète des modifications.
Enregistrement SAS	Enregistrement sécurisé et protégé cryptographiquement pouvant contenir des données arbitraires, notamment des données d'encaissement. Les Enregistrements SAS sont chaînés entre eux et signés ; le journal qu'ils forment ne permet que l'ajout — aucun enregistrement ne peut y être modifié, supprimé ni inséré rétroactivement. Chaque émission de justificatif y donne lieu à un enregistrement, distinguant l'original des duplicata.
Métier	Se réfère à toutes les activités quotidiennes d'une société. Une application métier est donc un logiciel qui traite des données métiers, c'est-à-dire des données liées aux clients, fournisseurs, produits, ventes, etc.
Proxy, proxy inverse, reverse proxy	Serveur mandataire qui s'occupe d'acheminer le trafic client externe vers un réseau interne et ses serveurs applicatifs.
Queue	File d'attente permettant d'exécuter des actions en tâche de fond. Chaque tâche s'exécute dans un contexte distinct : au regard de la traçabilité des justificatifs, une tâche constitue une émission (voir ce terme).
Émission	Remise d'un justificatif à un tiers, par impression ou par envoi. Une émission correspond à un contexte d'exécution unique du logiciel : une requête reçue du navigateur, ou une tâche d'arrière-plan. Le nombre de destinataires d'un même envoi ne multiplie pas les émissions ; une nouvelle tentative d'envoi constitue en revanche une émission distincte.
Original	Exemplaire issu de la première émission d'un justificatif validé. Il porte les mentions obligatoires, est archivé sous forme de fichier PDF nommé d'après son empreinte SHA-256, et donne lieu à un unique Enregistrement SAS d'impression d'original. Le fichier archivé n'est jamais modifié par la suite.

Duplicata	Exemplaire issu de toute émission postérieure à la première. Il est construit à partir de l'original archivé, relu et vérifié — jamais d'une nouvelle édition du document — puis annoté d'un filigrane « DUPLICATA », de son numéro d'ordre, de sa date de génération et de l'empreinte de l'original. Chaque duplicata donne lieu à un Enregistrement SAS. Le numéro d'ordre est continu et commun à tous les canaux d'émission d'un même justificatif.
Aperçu	Affichage d'un justificatif à l'écran, à des fins de vérification. L'aperçu n'est pas une émission : il ne donne lieu à aucun Enregistrement SAS. Le document présenté est recouvert de hachures et porte la mention « Ne pas imprimer. Ne pas envoyer. », ce qui le rend inexploitable comme justificatif.
Réimpression	Terme d'usage désignant toute émission postérieure à la première. Une réimpression produit toujours un duplicata ; le mot ne désigne jamais la restitution à l'identique d'un original.
Filigrane	Mention apposée en diagonale sur toutes les pages d'un document imprimé pour signaler sa nature : « DUPLICATA » pour un duplicata, mention de caractère provisoire pour un document non validé.
Périmètre fiscal	Ensemble des fichiers de code source assurant les fonctions de sécurisation, d'inaltérabilité, de conservation et d'archivage au sens du BOI-TVA-30-10-30. Sa composition est listée en annexe du dossier de maintenance ; son intégrité est garantie par l'empreinte de version.
Empreinte de version	Fichier publié avec chaque version de l'application, associant à chaque fichier du périmètre fiscal l'empreinte cryptographique de son contenu. Il constitue la preuve d'intégrité de la version publiée et permet à un auditeur de vérifier que le code déployé est bien celui qui a été évalué.
LNE	Laboratoire national de métrologie et d'essais. Organisme certificateur des systèmes de caisse au titre de l'article 88 de la loi n° 2015-1785, et éditeur du référentiel de certification applicable à Dokos.

Annexe 2 - Principaux Modèles Logiques de Données

DocType - Type de document

```sql
DROP TABLE IF EXISTS `tabDocType`;
CREATE TABLE `tabDocType` (
  `name` varchar(255) NOT NULL,
  `creation` datetime(6) DEFAULT NULL,
  `modified` datetime(6) DEFAULT NULL,
  `modified_by` varchar(255) DEFAULT NULL,
  `owner` varchar(255) DEFAULT NULL,
  `docstatus` tinyint NOT NULL DEFAULT 0,
  `idx` int NOT NULL DEFAULT 0,
  `search_fields` varchar(255) DEFAULT NULL,
  `issingle` tinyint NOT NULL DEFAULT 0,
  `is_virtual` tinyint NOT NULL DEFAULT 0,
  `is_tree` tinyint NOT NULL DEFAULT 0,
  `istable` tinyint NOT NULL DEFAULT 0,
  `editable_grid` tinyint NOT NULL DEFAULT 1,
  `track_changes` tinyint NOT NULL DEFAULT 0,
  `module` varchar(255) DEFAULT NULL,
  `restrict_to_domain` varchar(255) DEFAULT NULL,
  `app` varchar(255) DEFAULT NULL,
  `autoname` varchar(255) DEFAULT NULL,
  `naming_rule` varchar(40) DEFAULT NULL,
  `title_field` varchar(255) DEFAULT NULL,
  `image_field` varchar(255) DEFAULT NULL,
  `timeline_field` varchar(255) DEFAULT NULL,
  `sort_field` varchar(255) DEFAULT NULL,
  `sort_order` varchar(255) DEFAULT NULL,
  `description` text,
  `colour` varchar(255) DEFAULT NULL,
  `read_only` tinyint NOT NULL DEFAULT 0,
  `in_create` tinyint NOT NULL DEFAULT 0,
  `menu_index` int DEFAULT NULL,
  `parent_node` varchar(255) DEFAULT NULL,
  `smallicon` varchar(255) DEFAULT NULL,
```

```
  `allow_copy` tinyint NOT NULL DEFAULT 0,
  `allow_rename` tinyint NOT NULL DEFAULT 0,
  `allow_import` tinyint NOT NULL DEFAULT 0,
  `hide_toolbar` tinyint NOT NULL DEFAULT 0,
  `track_seen` tinyint NOT NULL DEFAULT 0,
  `max_attachments` int NOT NULL DEFAULT 0,
  `print_outline` varchar(255) DEFAULT NULL,
  `document_type` varchar(255) DEFAULT NULL,
  `icon` varchar(255) DEFAULT NULL,
  `color` varchar(255) DEFAULT NULL,
  `tag_fields` varchar(255) DEFAULT NULL,
  `subject` varchar(255) DEFAULT NULL,
  `_last_update` varchar(32) DEFAULT NULL,
  `engine` varchar(20) DEFAULT 'InnoDB',
  `default_print_format` varchar(255) DEFAULT NULL,
  `is_submittable` tinyint NOT NULL DEFAULT 0,
  `show_name_in_global_search` tinyint NOT NULL DEFAULT 0,
  `_user_tags` varchar(255) DEFAULT NULL,
  `custom` tinyint NOT NULL DEFAULT 0,
  `beta` tinyint NOT NULL DEFAULT 0,
  `has_web_view` tinyint NOT NULL DEFAULT 0,
  `allow_guest_to_view` tinyint NOT NULL DEFAULT 0,
  `route` varchar(255) DEFAULT NULL,
  `is_published_field` varchar(255) DEFAULT NULL,
  `website_search_field` varchar(255) DEFAULT NULL,
  `email_append_to` tinyint NOT NULL DEFAULT 0,
  `subject_field` varchar(255) DEFAULT NULL,
  `sender_field` varchar(255) DEFAULT NULL,
  `show_title_field_in_link` tinyint NOT NULL DEFAULT 0,
  `migration_hash` varchar(255) DEFAULT NULL,
  `translated_doctype` tinyint NOT NULL DEFAULT 0,
  PRIMARY KEY (`name`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

DocField

```sql
DROP TABLE IF EXISTS `tabDocField`;
CREATE TABLE `tabDocField` (
  `name` varchar(255) NOT NULL,
  `creation` datetime(6) DEFAULT NULL,
  `modified` datetime(6) DEFAULT NULL,
  `modified_by` varchar(255) DEFAULT NULL,
  `owner` varchar(255) DEFAULT NULL,
  `docstatus` tinyint NOT NULL DEFAULT 0,
  `parent` varchar(255) DEFAULT NULL,
  `parentfield` varchar(255) DEFAULT NULL,
  `parenttype` varchar(255) DEFAULT NULL,
  `idx` int NOT NULL DEFAULT 0,
  `fieldname` varchar(255) DEFAULT NULL,
  `label` varchar(255) DEFAULT NULL,
  `oldfieldname` varchar(255) DEFAULT NULL,
  `fieldtype` varchar(255) DEFAULT NULL,
  `oldfieldtype` varchar(255) DEFAULT NULL,
  `options` text,
  `search_index` tinyint NOT NULL DEFAULT 0,
  `show_dashboard` tinyint NOT NULL DEFAULT 0,
  `hidden` tinyint NOT NULL DEFAULT 0,
  `set_only_once` tinyint NOT NULL DEFAULT 0,
  `allow_in_quick_entry` tinyint NOT NULL DEFAULT 0,
  `print_hide` tinyint NOT NULL DEFAULT 0,
  `report_hide` tinyint NOT NULL DEFAULT 0,
  `reqd` tinyint NOT NULL DEFAULT 0,
  `bold` tinyint NOT NULL DEFAULT 0,
  `in_global_search` tinyint NOT NULL DEFAULT 0,
  `collapsible` tinyint NOT NULL DEFAULT 0,
  `unique` tinyint NOT NULL DEFAULT 0,
  `no_copy` tinyint NOT NULL DEFAULT 0,
  `allow_on_submit` tinyint NOT NULL DEFAULT 0,
  `show_preview_popup` tinyint NOT NULL DEFAULT 0,
  `trigger` varchar(255) DEFAULT NULL,
  `collapsible_depends_on` text,
  `mandatory_depends_on` text,
```

```
  `read_only_depends_on` text,
  `depends_on` text,
  `permlevel` int NOT NULL DEFAULT 0,
  `ignore_user_permissions` tinyint NOT NULL DEFAULT 0,
  `width` varchar(255) DEFAULT NULL,
  `print_width` varchar(255) DEFAULT NULL,
  `columns` int NOT NULL DEFAULT 0,
  `default` text,
  `description` text,
  `in_list_view` tinyint NOT NULL DEFAULT 0,
  `fetch_if_empty` tinyint NOT NULL DEFAULT 0,
  `in_filter` tinyint NOT NULL DEFAULT 0,
  `remember_last_selected_value` tinyint NOT NULL DEFAULT 0,
  `ignore_xss_filter` tinyint NOT NULL DEFAULT 0,
  `print_hide_if_no_value` tinyint NOT NULL DEFAULT 0,
  `allow_bulk_edit` tinyint NOT NULL DEFAULT 0,
  `in_standard_filter` tinyint NOT NULL DEFAULT 0,
  `in_preview` tinyint NOT NULL DEFAULT 0,
  `read_only` tinyint NOT NULL DEFAULT 0,
  `precision` varchar(255) DEFAULT NULL,
  `max_height` varchar(10) DEFAULT NULL,
  `length` int NOT NULL DEFAULT 0,
  `translatable` tinyint NOT NULL DEFAULT 0,
  `hide_border` tinyint NOT NULL DEFAULT 0,
  `hide_days` tinyint NOT NULL DEFAULT 0,
  `hide_seconds` tinyint NOT NULL DEFAULT 0,
  PRIMARY KEY (`name`),
  KEY `parent` (`parent`),
  KEY `label` (`label`),
  KEY `fieldtype` (`fieldtype`),
  KEY `fieldname` (`fieldname`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

DocPerm - Table de permissions par défaut

```sql
DROP TABLE IF EXISTS `tabDocPerm`;
CREATE TABLE `tabDocPerm` (
  `name` varchar(255) NOT NULL,
  `creation` datetime(6) DEFAULT NULL,
  `modified` datetime(6) DEFAULT NULL,
  `modified_by` varchar(255) DEFAULT NULL,
  `owner` varchar(255) DEFAULT NULL,
  `docstatus` tinyint NOT NULL DEFAULT 0,
  `parent` varchar(255) DEFAULT NULL,
  `parentfield` varchar(255) DEFAULT NULL,
  `parenttype` varchar(255) DEFAULT NULL,
  `idx` int NOT NULL DEFAULT 0,
  `permlevel` int DEFAULT '0',
  `role` varchar(255) DEFAULT NULL,
  `match` varchar(255) DEFAULT NULL,
  `read` tinyint NOT NULL DEFAULT 1,
  `write` tinyint NOT NULL DEFAULT 1,
  `create` tinyint NOT NULL DEFAULT 1,
  `submit` tinyint NOT NULL DEFAULT 0,
  `cancel` tinyint NOT NULL DEFAULT 0,
  `delete` tinyint NOT NULL DEFAULT 1,
  `amend` tinyint NOT NULL DEFAULT 0,
  `report` tinyint NOT NULL DEFAULT 1,
  `export` tinyint NOT NULL DEFAULT 1,
  `import` tinyint NOT NULL DEFAULT 0,
  `share` tinyint NOT NULL DEFAULT 1,
  `print` tinyint NOT NULL DEFAULT 1,
  `email` tinyint NOT NULL DEFAULT 1,
  PRIMARY KEY (`name`),
  KEY `parent` (`parent`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

Series - Préfixes de masque de numérotation

```sql
DROP TABLE IF EXISTS `tabSeries`;
CREATE TABLE `tabSeries` (
  `name` varchar(100),
  `current` int NOT NULL DEFAULT 0,
  PRIMARY KEY(`name`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

Sessions - Sessions des utilisateurs

```sql
DROP TABLE IF EXISTS `tabSessions`;
CREATE TABLE `tabSessions` (
  `user` varchar(255) DEFAULT NULL,
  `sid` varchar(255) DEFAULT NULL,
  `sessiondata` longtext,
  `ipaddress` varchar(16) DEFAULT NULL,
  `lastupdate` datetime(6) DEFAULT NULL,
  `status` varchar(20) DEFAULT NULL,
  KEY `sid` (`sid`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

Single - Singletons

```sql
DROP TABLE IF EXISTS `tabSingles`;
CREATE TABLE `tabSingles` (
  `doctype` varchar(255) DEFAULT NULL,
  `field` varchar(255) DEFAULT NULL,
  `value` longtext,
  KEY `singles_doctype_field_index` (`doctype`, `field`)
```

```
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

_Auth - Stockage des mots de passe utilisateurs

```
SQL
DROP TABLE IF EXISTS `__Auth`;
CREATE TABLE `__Auth` (
    `doctype` VARCHAR(140) NOT NULL,
    `name` VARCHAR(255) NOT NULL,
    `fieldname` VARCHAR(140) NOT NULL,
    `password` TEXT NOT NULL,
    `encrypted` tinyint NOT NULL DEFAULT 0,
    PRIMARY KEY (`doctype`, `name`, `fieldname`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

File - Structure de métadonnées de fichiers

```
SQL
DROP TABLE IF EXISTS `tabFile`;
CREATE TABLE `tabFile` (
  `name` varchar(255) NOT NULL,
  `creation` datetime(6) DEFAULT NULL,
  `modified` datetime(6) DEFAULT NULL,
  `modified_by` varchar(255) DEFAULT NULL,
  `owner` varchar(255) DEFAULT NULL,
  `docstatus` tinyint NOT NULL DEFAULT 0,
  `parent` varchar(255) DEFAULT NULL,
  `parentfield` varchar(255) DEFAULT NULL,
  `parenttype` varchar(255) DEFAULT NULL,
  `idx` int NOT NULL DEFAULT 0,
  `file_name` varchar(255) DEFAULT NULL,
  `file_url` varchar(255) DEFAULT NULL,
  `module` varchar(255) DEFAULT NULL,
```

```
  `attached_to_name` varchar(255) DEFAULT NULL,
  `file_size` int NOT NULL DEFAULT 0,
  `attached_to_doctype` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`name`),
  KEY `parent` (`parent`),
  KEY `creation` (`creation`),
  KEY `attached_to_name` (`attached_to_name`),
  KEY `attached_to_doctype` (`attached_to_doctype`)
) ENGINE=InnoDB ROW_FORMAT=DYNAMIC CHARACTER SET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```