Application de gestion du bureau d'ordre – Plan

---

1. Objectif & périmètre

Développer une application web permettant au Bureau d'ordre d'enregistrer chaque courrier entrant ou sortant — papier ou numérique — et de le suivre grâce aux numéros de texte de projet de loi.

---

2. Pile technologique (Tech Stack)

Couche	Outil	Pourquoi
Back-end	**Django 5 + Django REST Framework**	Framework complet, interface d'admin intégrée, grande communauté
Base de données	**PostgreSQL**	Gratuit, fiable, recherche textuelle puissante
Stockage de fichiers	Disque local (dev) → MinIO/S3 (prod)	Même code du test local au serveur
Tâches asynchrones	Celery + Redis (optionnel en Semaine 4)	Lance l'OCR & la génération de QR sans bloquer les requêtes
Front-end	Templates Django + HTMX (amélioration progressive)	N'est pas requis de mettre des pipelines JS séparé
OCR	Tesseract via *pytesseract*	Open-source, liaison Python simple
Code-barres / QR	*python-barcode*, *qrcode*	Bibliothèques simples, installables via pip
CI / Contrôle de version	GitHub + GitHub Actions	Tests automatiques & image Docker générée

Remarque : React, Kubernetes, micro-services, etc. sont réservés à l'après-stage.

---

3. Fonctionnalités minimales viables (priorité stagiaire)

3.1 Enregistrement de document

- Formulaire pour saisir les métadonnées des projets de loi.
- Génération d'un **accusé PDF** contenant un QR code embarqué.
- Sauvegarde des métadonnées dans la BD.

3.2 Recherche & liste

- Vue liste avec filtres simples (date d'enregistrement, secteur, titre etc….).
- Recherche plein-texte sur l'objet + le numéro de texte.

3.3 Scan de masse & OCR (objectif étiré ≤ 6 sem.)

- Téléversement de fichiers multiples ou d'un ZIP.
- Tâche exécutant Tesseract, stockant le texte extrait pour la recherche.

Optionnel (si le temps le permet) : notification e-mail à chaque nouvel enregistrement.

---

4. Feuille de route sur six semaines

Semaine	Tâches time-boxées
S1 – Mise en place & Quick Win	• Installer Python/Django • Créer le dépôt GitHub • Maquettage projet + admin • avoir une idée générale sur le processus métier d'enregistrement.
S2 – CRUD & Vue liste	• Formulaire de création • Template liste • Téléversement PDF unique • Style Bootstrap
S3 – Recherche & filtres	• Recherche basique • Widget filtre plage de dates
S4 – QR/Code-barres & accusé	• Générer PNG code-barres + QR à l'enregistrement • Créer le reçu PDF (ReportLab) • Tests unitaires
S5 – Téléversement groupé + OCR	• Zone de dépôt multi-fichiers • Worker Celery OCR • Afficher le texte extrait
S6 – Finition & passation	• Compteurs tableau de bord • Rédiger README + guide (≤ 5 pages) • Session de recette avec la responsable d'enregistrement.

---

5. Ressources d'apprentissage

Sujet	Lien
Tutoriel officiel Django	https://docs.djangoproject.com/en/5.0/intro/
Démarrage rapide DRF	https://www.django-rest-framework.org/tutorial/quickstart/
Exemples HTMX	https://htmx.org/examples/
Tesseract OCR en Python	https://nanonets.com/blog/ocr-with-tesseract/
Code-barres & QR en Python	https://pypi.org/project/python-barcode/ , https://pypi.org/project/qrcode/

---

7. Feuille de route post-stage

1. **Contrôle d'accès RBAC** (Django-Guardian ou Keycloak).
2. **Moteur de workflow** (django-fsm ou micro-service Camunda).

3. Intégration **courrier sortant & signature électronique** (*Barid E-Sign*).
4. **Analytique & BI** : export vers Power BI, KPI (temps moyen de traitement).
5. **Intégrations API** avec d'autres systèmes.
6. **UI mobile** ou SPA React.
7. **Haute disponibilité & montée en charge** : conteneurisation Kubernetes, OpenSearch.

---

8. Risques & atténuations (vue stagiaire)

Risque	Action
Sous-estimation du temps	Démo hebdo ; supprimer d'abord les objectifs étirés
Faible précision OCR	Limiter aux PDF clairs ; marquer l'OCR en bêta

---

9. Critères de réussite

- La responsable d'enregistrement, enregistre un nouveau document en **moins de 2 minutes**.
- La recherche renvoie l'élément correct en **< 1 seconde** sur le portable de dev.