

|  |  |  |
| --- | --- | --- |
| *Le futur se dessine aujourd’hui* | | |
| Cahier  Des Charges  Technique | | |
| Version V01  Direction Technique  Buildingmap | | |
| Résultat de recherche d'images pour "icon orange mail" | contact@futurmap.com | Date d’édition :  mardi 14 octobre 2025 | |
| Résultat de recherche d'images pour "icon orange website" | www.futurmap.com |

# **Information Générales**

**Titre du projet : ConfianceFmap**

**Demandeur(s) : Département Qualité**

**Date de la demande :**

**Date limite souhaitée :**

**Version du document : 1.1**

**Glossaire :**

* **NCE** : Non-Conformité Externe
* **NPS** : Net Promoter Score
* **CSAT** : Customer Satisfaction
* **JWT** : JSON Web Token
* **MinIO** : Stockage objet compatible S3
* **CRUD**: create, read, update,delete.
* BLOB :Binary Large Object.
* UUID : Universal Unique Identifier
* PK : Primary Key

# Objectifs Techniques

## Objet :

L’objectif principal est de développer une plateforme web responsive permettant la centralisation et la traçabilité des livraisons, la gestion des non-conformités externes et la mesure de la satisfaction client.

## Objectifs secondaires :

* Assurer la traçabilité des livraisons et des accusés.
* Automatiser les sondages NPS et CSAT.
* Gérer les non-conformités externes liées aux livraisons.
* Offrir un tableau de bord qualité interactif.
* Garantir la sécurité et la conformité ISO9001 des données.

## Périmètre fonctionnel :

1. **Gestion des livraisons :**

* Création et organisation : chaque livraison attachée à un projet.
* Upload multi-format : PDF, Excel, Word, images, modèles 3D, ZIP.
* Notifications email : livraison envoyée aux guests via email.
* Accusé de réception : génération automatique de fichier PDF horodaté.
* Versioning minimal : possibilité de versionner les livrables si nécessaire.

1. **Collecte des NCE et sondages**

* **Méthode** : réponse email automatisée via EmailService (parsing des réponses et pièces jointes).
* Guests ne doivent pas accéder directement à l’application.
* NCE : déclaration simple via réponse email, pièces justificatives attachées.
* Survey (NPS/CSAT) : réponse par score simple ou clic dans email, traitement automatisé côté serveur.
* Historique complet : liens entre NCE, Survey et livraisons.

1. **Reporting et tableau de bord**

* Indicateurs clés : nombre de livraisons, taux de NCE, score moyen CSAT/NPS.
* Filtres par période, projet, client, catégorie de NCE.
* Export Excel/PDF.

# Spécifications fonctionnelles

## Cas d’usage principaux

* **Guest(client)** : reçoit email, peut répondre NCE/sondage par email, joindre fichiers justificatifs.
* **User interne(producer)** : crée/modifie/supprime livraisons, gère NCE, suit reporting.
* **Administrateur** : gestion des utilisateurs et permissions, configuration globale.

## Flux utilisateurs

* Flux centralisé : **User → Notification email → Guest → EmailService → Survey/NCE**

## Gestion des rôles et permissions

* Admin : CRUD complet sur toutes entités.
* Qualité : CRUD livraisons, NCE, sondages.
* Producer: CRUD livraisons, upload fichiers.
* Guest : Réception email, réponse NCE/Survey via email.

# ARCHITECTURE TECHNIQUE

* **Frontend** : Next.js (React, TypeScript, TailwindCSS)
* **Backend** : FastAPI (Python 3.12)
* B**ase de données** : PostgreSQL
* **Stockage fichiers** : MinIO (compatible S3)
* **Conteneurisation** : Docker / Docker Compose
* **Déploiement** : Local
* **Authentification** : JWT
* **Microsoft Graph API : EmailService local** pour envoi et traitement des réponses guest

# SPÉCIFICATIONS TECHNIQUES

* API contract **: Swagger / OpenAPI avec endpoints détaillés.**
* Convention de code **: PEP8 (Python), ESLint (TypeScript).**
* Versioning Git **: branches main/dev, commits conventionnels.**
* CI/CD **: GitHub Actions pour tests et déploiement.**
* Tests **: Pytest, Jest, Cypress (E2E).**

# Base de données

* **User** : id, email, role, created\_at
* **Guest** : id, email, name, created\_at
* **Project** : id, name, owner\_id
* **Delivery** : id, project\_id, title, description, issuer\_id, created\_at, status, published\_at
* **File** : id, storage\_key, filename, delivery\_id, nce\_id, created\_at
* **Receipt** : id, delivery\_id, file\_id, generated\_by, generated\_at
* **Notification** : id, delivery\_id, to\_email, channel, status, sent\_at
* **Survey** : id, delivery\_id (nullable), type (NPS/CSAT), score, comment, respondent\_email, responded\_at
* **NCE** : id, delivery\_id, title, description, category, severity, status, created\_at, attachments
* **AuditLog** : id, user\_id, action\_type, entity\_type, entity\_id, timestamp

# SÉCURITÉ

* HTTPS.
* Authentification JWT.
* Gestion fine des rôles et permissions.
* Politique de mot de passe gérée par **Azure Active Directory** (Microsoft Graph avec OAuth2 )

# DÉPLOIEMENT ET INFRASTRUCTURE

* Architecture réseau documentée.
* Environnements : dev / staging / production.
* Orchestration Docker Compose.

# TESTS ET QUALITÉ

* Stratégie tests complète : unitaires, intégration, E2E, performance.
* Couverture ≥ 80%.
* Critères d’acceptation définis.

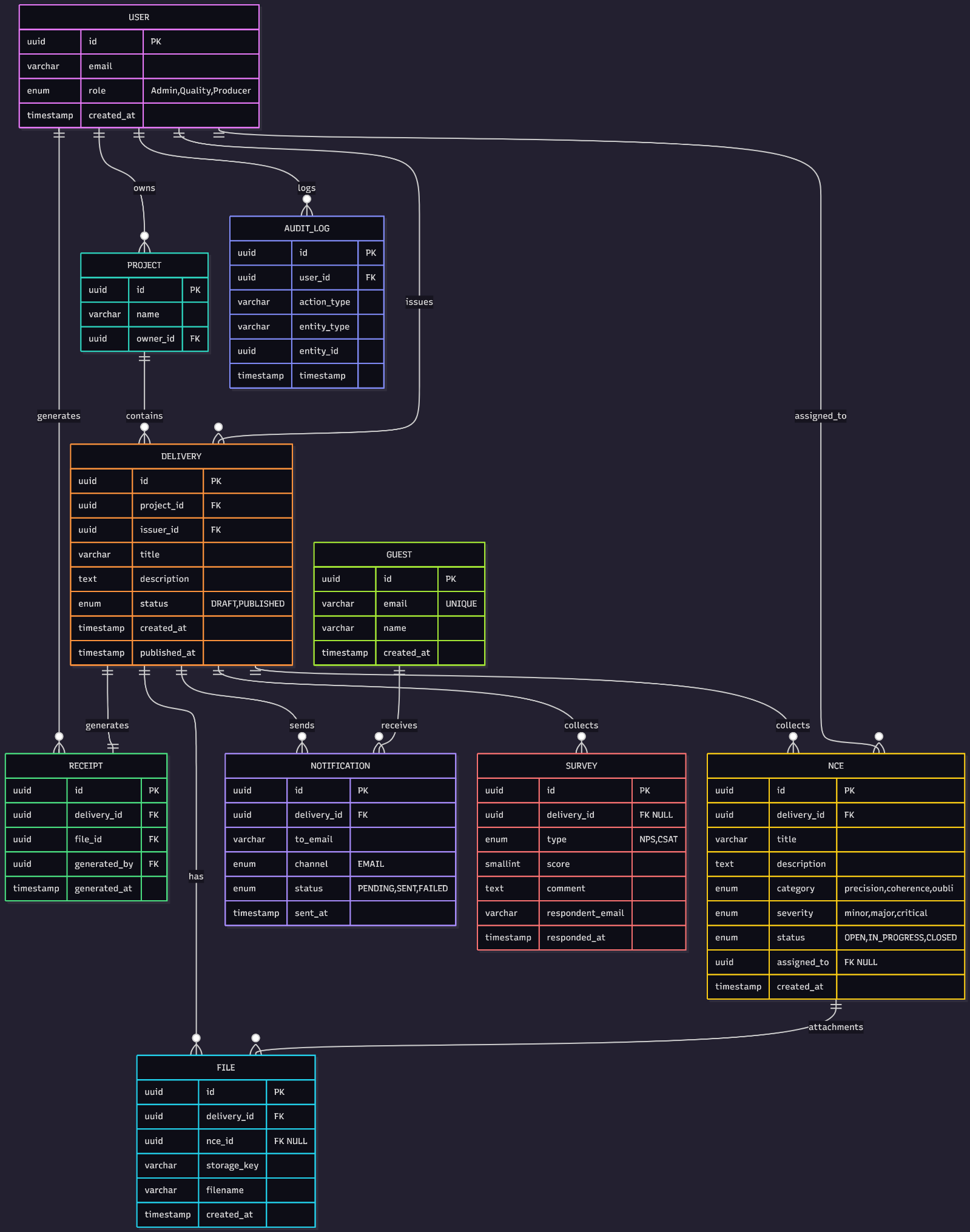
# LIVRABLES TECHNIQUES

* Code source complet.
* Documentation technique et API.
* Scripts Docker et configuration.
* Diagrammes architecture et BDD.
* Manuel d’installation.

# Annexes

* Diagrammes UML : cas d’usage, séquence.
* BDD: representation ERD (Entity-Relationship Diagram)
* Schéma d’architecture.
* Charte graphique.
* Exemples payload JSON pour API.

## Exemple de visualisation ERD de BDD



## Exemple de diagramme de classe :

