

**PROGRAMME DE FORMATION DÉVELOPPEUR EN INTELLIGENCE ARTIFICIELLE
OU CERTIFICATION PROFESSIONNELLE DÉVELOPPEUR EN INTELLIGENCE
ARTIFICIELLE ET DATA SCIENCE RNCP36581****Bloc E6.3 – Produire et maintenir une solution I.A**

Cahier des Charges de la MSPR « Mise en production d'une solution I.A à partir des éléments de contextes fournis (Cahier des charges - données d'entrée) »

COMPÉTENCES ÉVALUÉES :**Certification professionnelle Développeur en intelligence artificielle et data science RNCP36581 :**

- Développer le back-end : Développer les composants de la solution IA sous forme d'API et/ou des programmes intégrés en utilisant des outils adaptés.
- Développer le front-end : Développer l'interface homme-machine en utilisant les techniques, les outils et les plateformes dans l'objectif de rendre l'approche ergonomique et conforme à l'accessibilité numérique.
- Mettre en œuvre des plans de tests définis pour préparer le déploiement de la solution I.A.
- Superviser le fonctionnement de la solution IA à partir des outils de monitoring afin de détecter et corriger les éventuels dysfonctionnements dans une démarche d'amélioration continue.
- Corriger les dysfonctionnements de son périmètre de responsabilité.
- Réaliser les évolutions fonctionnelles de la solution I.A afin de répondre au besoin d'amélioration.

Programme de formation Développeur en intelligence artificielle :

- Analyser le besoin d'application d'un commanditaire intégrant un service d'intelligence artificielle, en rédigeant les spécifications fonctionnelles et en le modélisant, dans le respect des standards d'utilisabilité et d'accessibilité, afin d'établir avec précision les objectifs de développement correspondant au besoin et à la faisabilité technique.
- Concevoir le cadre technique d'une application intégrant un service d'intelligence artificielle, à partir de l'analyse du besoin, en spécifiant l'architecture technique et applicative et en préconisant les outils et méthodes de développement, pour permettre le développement du projet.
- Coordonner la réalisation technique d'une application d'intelligence artificielle en s'intégrant dans une conduite agile de projet et un contexte MLOps et en facilitant les temps de collaboration dans le but d'atteindre les objectifs de production et de qualité.
- Développer les composants techniques et les interfaces d'une application en utilisant les outils et langages de programmation adaptés et en respectant les spécifications fonctionnelles et techniques, les standards et normes d'accessibilité, de sécurité et de gestion des données en vigueur dans le but de répondre aux besoins fonctionnels identifiés.
- Automatiser les phases de tests du code source lors du versionnement des sources à l'aide d'un outil d'intégration continue de manière à garantir la qualité technique des réalisations.
- Créer un processus de livraison continue d'une application en s'appuyant sur une chaîne d'intégration continue et en paramétrant les outils d'automatisation et les environnements de test afin de permettre une restitution optimale de l'application.
- Surveiller une application d'intelligence artificielle, en mobilisant des techniques de monitoring et de journalisation, dans le respect des normes de gestion des données personnelles en vigueur, afin d'alimenter la feedback loop dans une approche MLOps, et de permettre la détection automatique d'incidents
- Résoudre les incidents techniques en apportant les modifications nécessaires au code de l'application et en documentant les solutions pour en garantir le fonctionnement opérationnel.

PHASE 1 : PRÉPARATION DE CETTE MISE EN SITUATION PROFESSIONNELLE RECONSTITUÉE

- **Durée de préparation** : 19 heures
- **Mise en œuvre** : Travail d'équipe constituée de 4 apprenants (5 maximum si groupe impair)

PHASE 2 : PRÉSENTATION ORALE COLLECTIVE + ENTRETIEN COLLECTIF

- **Objectif** : mettre en avant et démontrer que les compétences visées par ce bloc sont bien acquises.
- **Moyen** : L'équipe utilise un support de présentation
- **Durée totale par groupe** : 50 mn se décomposant comme suit :
 - 20 mn de soutenance orale par l'équipe.
 - 30 mn d'entretien collectif avec le jury (questionnement complémentaire).
- **Jury d'évaluation** : 2 personnes (binôme d'évaluateurs) par jury – Ces évaluateurs ne sont pas intervenus durant la période de formation et ne connaissent pas les apprenants à évaluer.

I – CONTEXTE

ObRail Europe est un observatoire indépendant spécialisé dans le ferroviaire et la mobilité durable. Créée en 2018, l'organisation s'est imposée comme un acteur de référence dans l'analyse des flux ferroviaires européens et la promotion du transport bas-carbone.

Sa mission principale est de :

- Collecter et analyser les données relatives aux dessertes ferroviaires en Europe.
- Évaluer l'impact environnemental des modes de transport longue distance.
- Produire des études comparatives destinées aux décideurs politiques et aux opérateurs ferroviaires.
- Accompagner la transition écologique en favorisant l'usage du train comme alternative à l'avion sur les trajets intra-européens.

ObRail travaille en partenariat avec :

- Les institutions européennes (Commission européenne, Parlement européen) pour alimenter les politiques publiques.
- Des ONG environnementales telles que Transport & Environnement ou Back-on-Track afin de renforcer l'argumentaire écologique.
- Les opérateurs ferroviaires (SNCF, ÖBB Nightjet, DB, Trenitalia) qui souhaitent valoriser leurs services de trains de jour et de nuit.

L'organisation s'inscrit dans les grandes stratégies européennes :

- Le **Green Deal européen**, qui vise la neutralité carbone à l'horizon 2050.
- Le programme TEN-T (Trans-European Transport Network), qui a pour objectif de renforcer l'intégration et la compétitivité du réseau ferroviaire en Europe.

ObRail souhaite comparer la contribution des trains de jour et des trains de nuit au maillage ferroviaire européen. L'objectif est de **mesurer leur rôle dans la construction d'une mobilité durable** et d'évaluer leur potentiel en tant qu'alternative crédible à l'avion sur les trajets intra-européens.

Pour cela, il est nécessaire de constituer un référentiel de données fiable et harmonisé qui servira :

- à l'analyse de la couverture ferroviaire,
- à l'entraînement de modèles d'IA,
- et au développement d'un service applicatif destiné aux partenaires.

L'étude préliminaire réalisée par ObRail Europe a mis en évidence plusieurs contraintes majeures auxquelles le projet devra répondre :

1. Dispersion des données

Les informations relatives aux dessertes ferroviaires européennes sont publiées par une multitude d'acteurs (opérateurs nationaux, plateformes open data, organismes statistiques). Chaque source utilise son propre format (CSV, flux GTFS, API, fichiers Excel, HTML scraping), ce qui rend leur intégration particulièrement complexe.

2. Qualité hétérogène des jeux de données

Les données collectées présentent des problèmes de complétude et de fiabilité : doublons, informations manquantes (codes de gares, fuseaux horaires, identifiants de lignes), incohérences dans les unités ou les formats de date/heure. Ces particularités doivent être traitées en amont afin de garantir la robustesse des analyses futures.

3. Absence de standardisation transfrontalière

Il n'existe pas de référentiel commun permettant de comparer facilement les dessertes ferroviaires entre pays européens. Chaque opérateur applique ses propres conventions (nomenclature des gares, structure des horaires, indicateurs de performance), ce qui empêche une comparaison homogène des données.

4. Exigences réglementaires

Le projet doit se conformer au Règlement Général sur la Protection des Données (RGPD). Même si aucune donnée personnelle n'est traitée, les règles de sécurité, de documentation et de transparence dans la gestion des jeux de données doivent être respectées.

5. Contraintes temporelles

ObRail doit livrer une première version exploitable de l'entrepôt de données aux institutions européennes avant la fin de l'année. Cette échéance impose un processus ETL fiable, automatisé et reproductible, permettant de garantir la mise à jour régulière des données

II – CAHIER DES CHARGES

ObRail Europe dispose désormais d'un premier socle technique constitué :

- d'un entrepôt de données harmonisé permettant d'explorer les dessertes ferroviaires européennes,
- et d'une API REST prototype assurant un accès initial aux données.

Ces éléments, issus de la précédente mise en situation (TPRE512), représentent une base fonctionnelle de la future solution d'intelligence artificielle destinée à modéliser la contribution du rail à la mobilité durable mais encore insuffisante pour être utilisée en production.

À ce stade, la solution reste au niveau de **prototype non industrialisé** :

- le déploiement reste manuel,
- les tests ne sont pas automatisés,
- aucune supervision n'assure la disponibilité ou la qualité du service.

Afin de garantir la pérennité, la qualité technique et la supervision opérationnelle de la solution, ObRail Europe souhaite désormais industrialiser et mettre en production l'application. Cette étape vise à préparer l'intégration future du modèle d'IA tout en livrant une **application web complète, conteneurisée, testée et supervisée**, destinée à ses partenaires institutionnels et opérationnels.

Objectif général du projet

Mettre en production une solution applicative complète – composée d'un backend et d'un frontend - exploitant les données, intégrant une chaîne d'intégration et de livraison continues (CI/CD), et permettant la supervision en temps réel de son fonctionnement.

Cette solution doit être prête à accueillir un modèle d'intelligence artificielle lors des phases ultérieures du projet.

Objectifs opérationnels

Pour atteindre cet objectif général, les équipes devront répondre aux attentes suivantes :

1. Industrialisation de la solution existante

Stabiliser le backend et renforcer son architecture.

Conteneuriser l'ensemble de l'application (backend, frontend, base de données, monitoring).

Garantir la reproductibilité du déploiement via Docker et/ou Docker Compose.

2. Développement d'une interface utilisateur professionnelle

Concevoir et implémenter un frontend ergonomique, accessible et conforme aux standards Web.

Permettre la consultation, la recherche et la visualisation de statistiques ferroviaires.

Offrir une expérience adaptée aux besoins des partenaires d'ObRail (ONG, institutions, opérateurs).

3. Mise en œuvre d'une stratégie de tests complète

Développer et documenter les tests unitaires et d'intégration du backend.

Implémenter des tests end-to-end (E2E) sur le frontend via des outils adaptés (Cypress, Playwright...).

Garantir une couverture de tests suffisante pour la mise en production.

4. Automatisation du processus de déploiement (CI/CD)

Définir, implémenter et documenter un pipeline CI/CD (GitHub Actions, GitLab CI ou Jenkins).

Automatiser :

- l'exécution des tests,
- la création des images Docker,
- la mise à disposition de versions testables.

5. Mise en place d'une supervision et d'une observabilité

Installer un outil de monitoring (Grafana, Prometheus, Metabase ou équivalent).

Suivre les métriques essentielles : état/latence API, taux d'erreurs, logs applicatifs, anomalies.

Proposer un tableau de bord opérationnel pour les équipes internes d'ObRail.

6. Respect des exigences légales et techniques

Mise en conformité avec le RGPD et les bonnes pratiques de développement responsable.

Garantie de l'accessibilité numérique du frontend (RGAA).

Respect des normes de sécurité (gestion des erreurs, filtrage des entrées, logs).

7. Rapport technique complet

Produire un rapport technique complet incluant architecture, pipeline CI/CD, monitoring et tests.

ObRail Europe attend une solution stable, reproductible, testée, supervisée et documentée, suffisamment robuste pour être livrée à ses parties prenantes.

L'application produite doit :

- être techniquement fiable,
- répondre à des usages réels,
- et constituer un socle solide pour l'intégration future du modèle d'IA.

Cette étape marque une transition entre la construction du socle de données (MSPR TPRES12) et la future exploitation via l'IA et les services applicatifs avancés (MSPR TPRES22).

III – LES BESOINS EXPRIMÉS OBRIL EUROPE

Pour garantir la mise en production d'une solution fiable, maintenable et conforme aux attentes du cahier des charges, ObRail Europe exige que l'équipe projet réponde à un ensemble de besoins fonctionnels, techniques et réglementaires. Ces besoins définissent précisément les éléments à concevoir, développer, tester et documenter.

1. Analyse et conception de l'architecture

ObRail attend que l'équipe définisse une architecture technique complète et cohérente, intégrant l'ensemble des composants nécessaires à l'application finale.

Cette architecture doit inclure :

- un **backend** assurant l'accès aux données ferroviaires harmonisées,
- un **frontend** ergonomique destiné aux utilisateurs institutionnels et associatifs,
- une **base de données** permettant la persistance des informations,
- un module de **supervision et monitoring**,
- une **chaîne CI/CD** automatisant tests, builds et déploiements,
- un environnement d'exécution reproductible basé sur Docker.

L'équipe devra fournir les **diagrammes d'architecture et d'interconnexion** permettant de visualiser clairement les flux, les dépendances et l'organisation des services.

2. Développement du backend

Le backend, déjà amorcé lors de la MSPR précédente, doit être finalisé, renforcé et mis en conformité avec un niveau de qualité attendu en production.

Le besoin porte notamment sur :

- la finalisation et la fiabilisation des endpoints principaux :
 - /trajets
 - /trajets/{id}
 - /stats/volumes
 - /health
- la mise en œuvre d'une **gestion des erreurs maîtrisée** (validation, statuts HTTP, messages clairs),
- la sécurisation basique du service,
- la création d'une documentation automatique type **OpenAPI / Swagger**,
- la réalisation de **tests unitaires et d'intégration**, exécutables automatiquement dans la chaîne CI/CD.

3. Développement du frontend

ObRail souhaite disposer d'une **interface web professionnelle et accessible, adaptée** à un public non technique mais exigeant.

L'application devra permettre :

- la consultation et le filtrage des trajets ferroviaires,
- l'affichage d'indicateurs clés (répartition jour/nuit, volumes par opérateur, etc.),
- la visualisation de l'état de santé du service et des éventuels incidents détectés par le monitoring,
- une navigation fluide et intuitive, respectant les recommandations d'accessibilité (RGAA).

L'équipe devra également mettre en place des tests end-to-end (E2E) à l'aide d'outils tels que Cypress ou Playwright.

4. Conteneurisation et orchestration

Pour assurer la reproductibilité et la fiabilité du déploiement, ObRail demande la conteneurisation complète de la solution.

Cela inclut :

- la création de **Dockerfile** pour le backend, le frontend et les services complémentaires,
- l'intégration d'un **docker-compose.yml** permettant l'exécution coordonnée de l'ensemble des services (application + base de données + monitoring),
- la garantie de la **persistance des données** et de la configuration,
- la capacité pour un évaluateur de relancer l'application en un seul commande.

5. Automatisation et pipeline CI/CD

Une chaîne d'intégration et de livraison continues doit être mise en place pour automatiser les étapes nécessaires au maintien de la qualité.

ObRail attend :

- un pipeline complet (GitHub Actions, GitLab CI ou Jenkins) incluant :
 - installation des dépendances,
 - exécution automatique des tests,
 - création des images Docker,
 - déploiement sur environnement de test,
- une documentation claire des fichiers YAML ou Jenkinsfile,
- la gestion correcte des secrets et variables d'environnement (selon les bonnes pratiques).

L'objectif est de disposer d'un **processus industrialisé, documenté, et reproductible**.

6. Supervision, observabilité et journalisation

ObRail veut une solution **monitorée en continu**, capable de détecter automatiquement incidents, anomalies et comportements inattendus.

Le besoin couvre :

- l'intégration d'un outil de monitoring (Grafana ou équivalent),
- l'affichage de métriques clés :
 - disponibilité de l'API,
 - latence,
 - taux d'erreurs,
 - logs applicatifs,
 - volumes de données,
- la mise en œuvre d'une politique de logs cohérente, permettant le diagnostic rapide d'incidents,
- une interface de supervision accessible aux équipes internes.

7. Documentation et maintenance

Enfin, ObRail exige une **documentation professionnelle complète**, comprenant :

- la description de l'architecture et des choix techniques,
- la démarche de mise en production,
- le détail du pipeline CI/CD,
- les procédures d'installation et d'exécution,
- les mesures de conformité RGPD,
- les éléments de supervision,
- et un **plan de maintenance / rollback** clair.

IV – LES LIVRABLES

À l'issue de la mission, ObRail Europe attend de l'équipe projet la remise d'un ensemble de livrables exécutables, documentés et directement exploitables.

Ces livrables doivent démontrer la capacité de l'équipe à industrialiser, déployer, tester, superviser et maintenir une solution applicative intégrant des composants d'intelligence artificielle.

L'ensemble des livrables doit pouvoir être évalué sans intervention extérieure, et l'application doit être exécutable sur un environnement standard via les fichiers fournis.

1. Application complète exécutable via Docker Compose

L'équipe doit remettre une solution entièrement conteneurisée, comprenant :

- le **backend** (API REST),
- le **frontend** (interface utilisateur),
- la **base de données**,
- l'outil de **monitoring**,
- tout service complémentaire nécessaire au fonctionnement.

L'ensemble doit être orchestré via un fichier docker-compose.yml, permettant à un évaluateur :

- de lancer l'application complète en une seule commande,
- d'accéder aux interfaces,
- de vérifier la persistance et la cohérence des données.

La conteneurisation doit être **documentée**, reproductible et conforme aux bonnes pratiques de déploiement.

2. Ensemble des Dockerfiles et fichiers de configuration

Pour chaque service de l'application, l'équipe devra fournir :

- un **Dockerfile** conforme aux standards de sécurité et de performance,
- un fichier **docker-compose.yml** correctement structuré,
- les instructions nécessaires pour la configuration et l'exécution.

Chaque fichier devra être lisible, commenté si nécessaire, et conforme aux préconisations du pipeline CI/CD.

3. Pipeline CI/CD fonctionnel

L'équipe devra livrer un pipeline d'intégration et de livraison continues complet, sous la forme d'un ou plusieurs fichiers YAML ou Jenkinsfile.

Ce pipeline devra impérativement inclure :

- la construction automatique des images Docker,
- l'exécution des tests unitaires, d'intégration et/ou E2E,
- la vérification de la qualité du code (si applicable),
- la mise à disposition automatique de versions testables,
- la documentation des variables d'environnement et secrets nécessaires.

L'évaluateur doit pouvoir relire, comprendre et exécuter ce pipeline sans ambiguïté.

4. Tests automatisés (unitaires, intégration, E2E)

L'équipe doit produire un système de tests permettant de garantir la stabilité de la solution.

Ces tests pourront inclure :

- **tests unitaires** sur les composants du backend,
- **tests d'intégration** pour vérifier l'interopérabilité backend / base de données,
- **tests end-to-end (E2E)** sur le frontend, réalisés via Cypress, Playwright ou équivalent.

Les tests doivent être :

- exécutables en local,
- intégrés à la CI/CD,
- accompagnés d'instructions claires,
- suffisamment représentatifs pour vérifier la robustesse de la solution.

5. Tableau de bord de supervision

L'équipe doit fournir un outil de supervision fonctionnel permettant de suivre en temps réel :

- la disponibilité de l'API,
- la latence des réponses,
- le taux d'erreurs,
- les métriques importantes du système,
- les logs applicatifs,
- et toute anomalie détectée.

La solution de monitoring (Grafana, Metabase, autre...) doit être accessible via Docker et documentée.

L'évaluateur doit pouvoir accéder au tableau de bord et vérifier son bon fonctionnement.

6. Rapport technique complet

Un rapport clair, structuré et professionnel devra être produit. Il constituera le document de référence du projet. Le rapport devra au minimum contenir :

- l'architecture globale de la solution (schémas exigés),
- les choix techniques argumentés (frameworks, outils, standards),
- la description du pipeline CI/CD,
- les résultats des tests et leur couverture,
- les éléments relatifs à la supervision,
- les procédures d'installation et d'exécution,
- les mesures mises en œuvre pour garantir :
 - **la conformité RGPD,**
 - **la sécurité,**
 - **l'accessibilité,**
- la stratégie de maintenance : mise à jour, correctifs, rollback.

Ce rapport constitue un livrable central pour l'évaluation.

7. Support de soutenance

En complément des livrables techniques, l'équipe projet devra préparer un support de présentation destiné à la soutenance finale devant le client (public technique). Ce support devra synthétiser les principaux éléments du travail réalisé : démarche suivie, difficultés rencontrées, solutions mises en place, résultats obtenus et perspectives.

Il est important de souligner que l'évaluation de cette MSPR repose sur la **combinaison des trois éléments suivants** :

- la qualité du **travail réalisé** au cours du projet,
- la pertinence et l'exhaustivité des **livrables remis** (scripts, modèles, base de données, API, documentation),
- et la capacité de l'équipe à **présenter, justifier et valoriser** ce travail lors de la soutenance orale.

Les équipes devront donc s'assurer que la soutenance reflète bien l'ensemble des compétences attendues, en démontrant à la fois la maîtrise technique et la capacité à communiquer efficacement auprès d'un client professionnel.

V – RESSOURCES FOURNIES

Afin de garantir la bonne réalisation de cette mission, ObRail Europe met à disposition de l'équipe projet un ensemble de ressources techniques, documentaires et organisationnelles.

Ces ressources constituent un **point d'appui**, mais ne dispensent en aucun cas l'équipe de mener ses propres recherches, choix technologiques et arbitrages techniques.

Les apprenants doivent donc les considérer comme des **matériaux de référence**, à intégrer dans leur démarche de conception et d'industrialisation.

1. Base PostgreSQL et API prototype

Selon le besoin, l'encadrant pédagogique pourra autoriser une équipe à reprendre une base issue du travail réel de la MSPR précédente, afin de garantir la continuité du projet fil rouge.

La première version de l'API REST réalisée précédemment peut être réutilisée, améliorée ou entièrement refactorisée selon les choix de l'équipe.

Les apprenants pourront s'appuyer des sources open data ferroviaires déjà exploitées précédemment.

2. Environnement et outils recommandés

ObRail Europe laisse une grande liberté d'outillage, mais recommande l'usage des technologies standards du secteur pour garantir la compatibilité et la maintenabilité du projet.

Langages et bibliothèques

- Python : pandas, requests, SQLAlchemy, FastAPI
- JavaScript / TypeScript : React, Vue.js, Node.js
- Tests : Pytest, Jest, Cypress, Playwright

Conteneurisation et orchestration

- Docker
- Docker Compose

Base de données

- PostgreSQL (ou équivalent compatible)

CI/CD

- GitHub Actions
- GitLab CI
- Jenkins

Ces outils ne sont pas imposés, mais tout choix doit être argumenté et documenté dans le rapport technique.

3. Assistance et périmètre

Dans le cadre de ce projet pédagogique, l'équipe projet n'aura aucun contact direct avec ObRail Europe. Le cahier des charges constitue la seule expression officielle du besoin. Toute demande de clarification devra être traitée avec l'encadrant pédagogique, jouant le rôle du client.

Les ressources mises à disposition ne constituent pas un projet clé en main. ObRail attend que les apprenants :

- enrichissent, améliorent et renforcent les composants fournis,
- prennent des décisions techniques argumentées,
- produisent une solution industrialisée qui dépasse largement l'état initial du prototype.

L'objectif pédagogique est que les apprenants démontrent leur capacité à passer d'un environnement semi-fonctionnel à une solution réellement industrialisée et maintenable.

4. Webographie

ObRail met à disposition une sélection de ressources officielles pour aider l'équipe à concevoir, tester, déployer et superviser la solution.

CI/CD et automatisation

- Configuration CI/CD (GitHub Actions, GitLab CI) : GitHub Actions — documentation officielle <https://docs.github.com/en/actions>
- Workflow Python CI/CD : <https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-python>
- GitLab CI/CD — guide officiel : <https://docs.gitlab.com/ee/ci/>
- Pipeline multi-stage (build → test → deploy) : https://docs.gitlab.com/ee/ci/pipelines/pipeline_architectures.html

Conteneurisation

- Introduction et premiers pas : <https://docs.docker.com/get-started/>
- Guide Docker Compose : <https://docs.docker.com/compose/>
- Bonnes pratiques Docker pour développeurs : <https://docs.docker.com/develop/dev-best-practices/>

Développement backend / frontend

- Frameworks de développement (backend / frontend) : FastAPI – framework Python moderne et performant : <https://fastapi.tiangolo.com/>
- React.js – documentation officielle : <https://react.dev/>
- Vue.js – guide officiel : <https://vuejs.org/guide/introduction.html>

Supervision et monitoring

- Metabase – outil open-source pour créer des tableaux de bord : <https://www.metabase.com/docs/latest/>
- Grafana – monitoring et visualisation de métriques : <https://grafana.com/docs/>
- Streamlit – alternative simple pour visualiser les données en Python : <https://docs.streamlit.io/>

Sécurité, RGPD et accessibilité

- Bonnes pratiques RGPD et développement responsable – CNIL : <https://www.cnil.fr/fr/developper-un-projet-respectueux-de-la-vie-privee>
- Accessibilité web – RGAA (référentiel officiel français) : <https://accessibilite.numerique.gouv.fr/methode/>

Ces ressources permettent d'assurer que le projet s'appuie sur des pratiques professionnelles actualisées.