# PROJET PFS : NEXT-GEN DEVSECOPS

**Plateforme Intelligente de Génération et Validation Automatisées de Pipelines CI/CD Sécurisés**


**Présenté par:**

- Sayf Eddine Laamri
- Oussama Bagy

**Encadré par:**

- Pr. Wissam Abbass


Soutenu le 22 Mai 2026

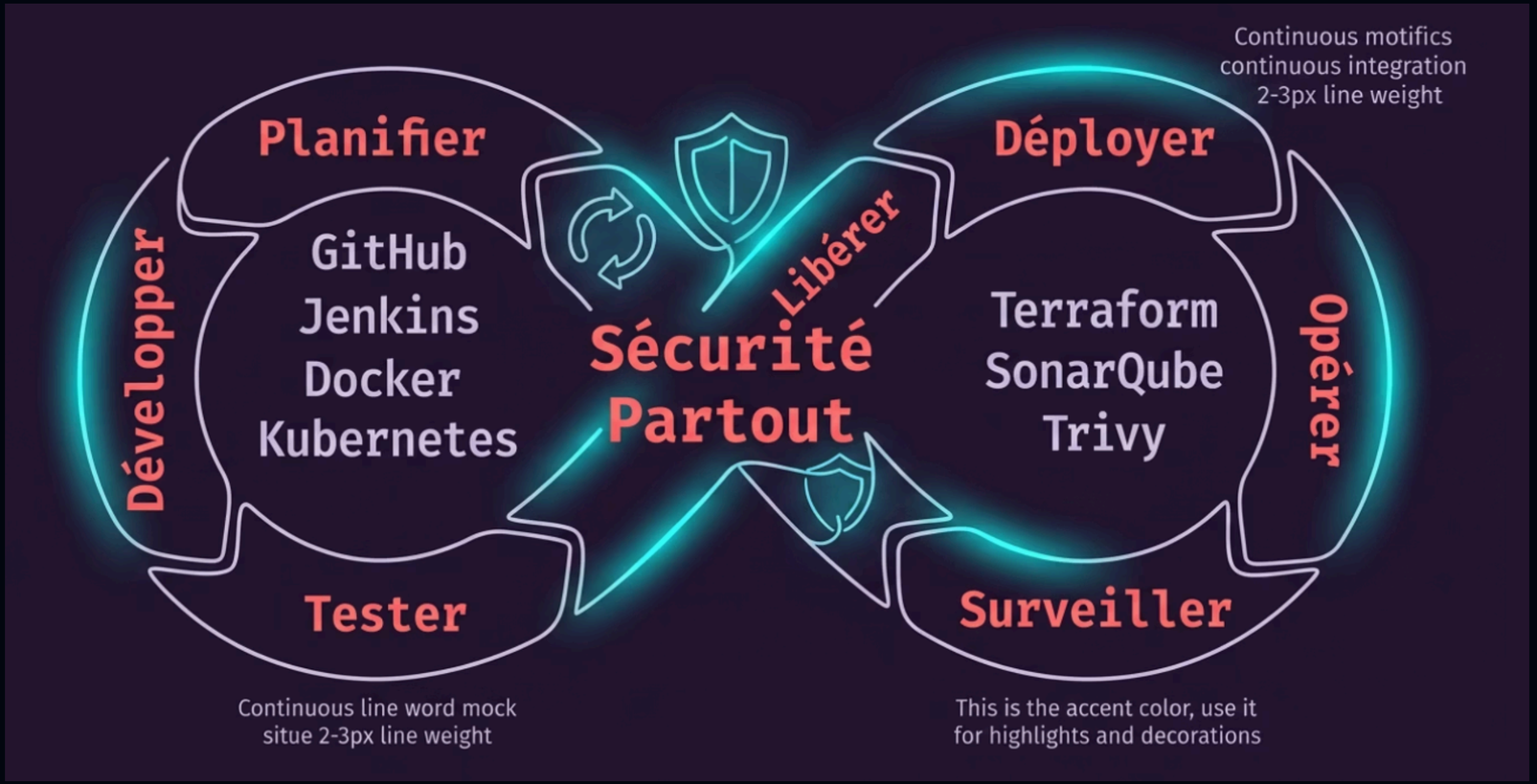
# REMERCIEMENTS

Nous remercions sincèrement le Pr. Wissam Abbass et le Pr. Loubna Alhaloui pour leur encadrement, leurs conseils techniques et leur disponibilité. Leur expertise a enrichi nos connaissances en cybersécurité, DevSecOps et automatisation.

# PLAN DE LA PRÉSENTATION

# QU'EST-CE QUE LE DEVSECOPS ?

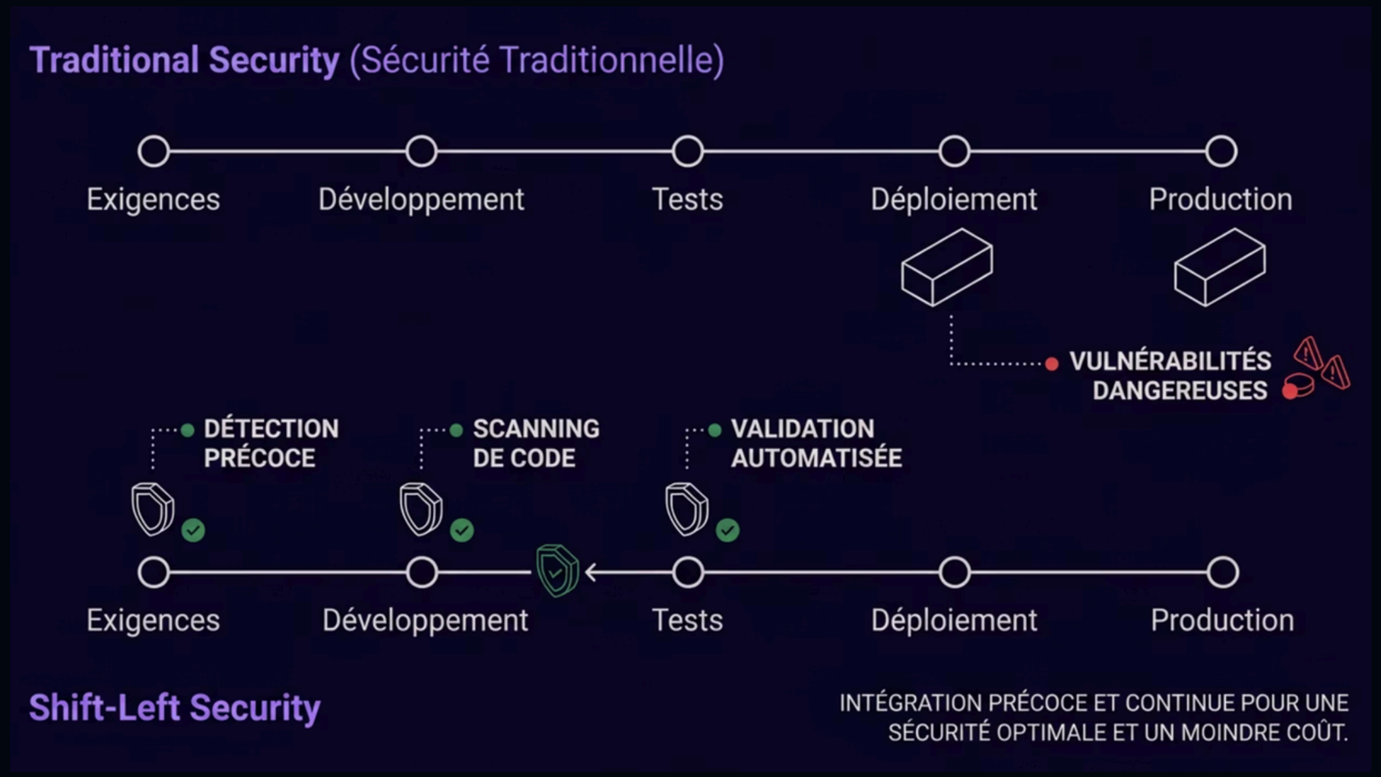
Le DevSecOps étend les principes du DevOps en intégrant la sécurité à chaque étape du cycle de vie du développement logiciel, de la planification à la surveillance continue.



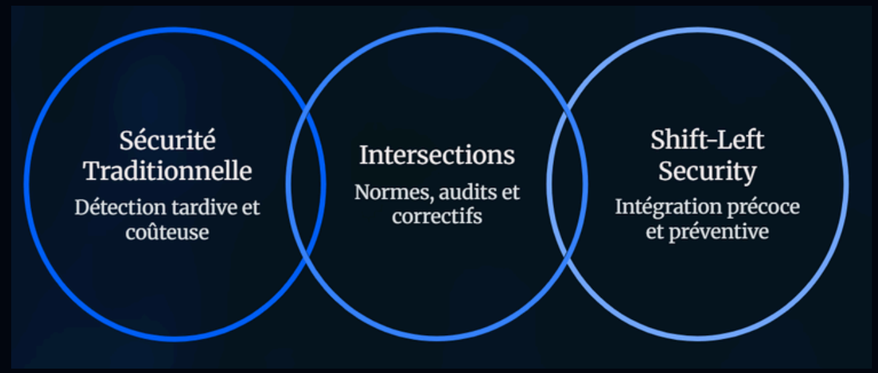

Le DevSecOps intègre la sécurité à chaque étape du développement et du déploiement, transformant la sécurité d'une réflexion après-coup en une partie intrinsèque du processus.

# SHIFT-LEFT SECURITY

Le principe du "Shift-Left Security" est fondamental en DevSecOps, visant à intégrer la sécurité dès les premières phases du développement logiciel.




## La sécurité déplacée vers la gauche




**« Détecter les vulnérabilités tôt réduit les risques, les coûts et les échecs de déploiement. »**

# CI/CD – CONTINUOUS INTEGRATION & CONTINUOUS DEPLOYMENT

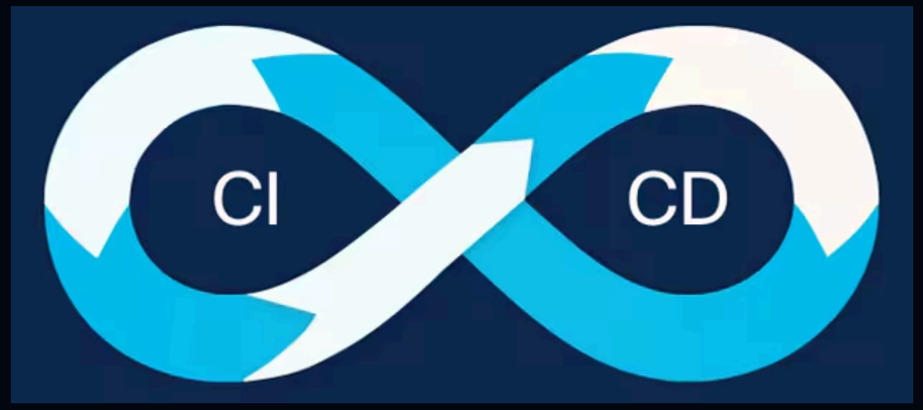
Le CI/CD est un ensemble de pratiques essentielles qui automatisent l'intégration, les tests et le déploiement d'une application, garantissant ainsi une livraison logicielle rapide et fiable.

## Intégration Continue (CI)

Intégration régulière du code dans un dépôt central (GitHub) et exécution automatique des tests afin de détectant les problèmes tôt.

## Déploiement Continu (CD)

Automatisation de la livraison des applications vers les environnements de test, pré-production ou production, une fois validées.



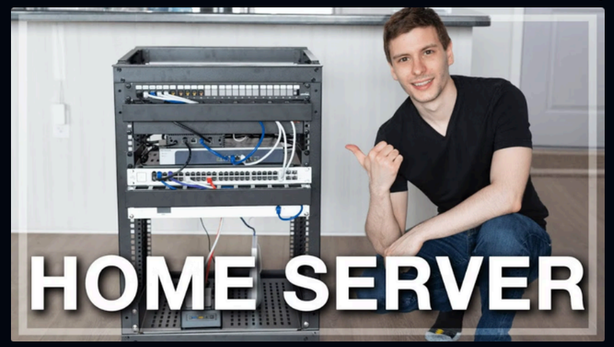

Bénéfices clés : réduction significative des erreurs manuelles, accélération des cycles de livraison et amélioration continue de la qualité logicielle.

# INFRASTRUCTURE AS CODE (IAC)

L'Infrastructure as Code (IaC) est une pratique clé du DevSecOps, permettant de gérer et provisionner l'infrastructure de manière automatisée, fiable et sécurisée.

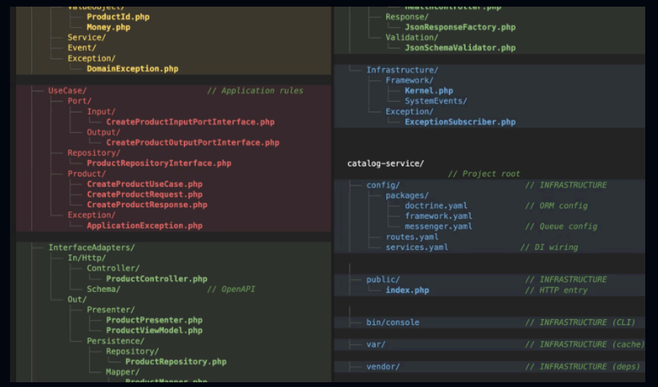
## PROBLÈME CLASSIQUE

La création manuelle des serveurs, réseaux et règles de sécurité conduit souvent à des erreurs, des lenteurs et une impossibilité de reproduire les environnements de manière identique.




## SOLUTION IAC

L'infrastructure est décrite sous forme de code, ce qui la rend reproductible, versionnable et contrôlable,




L'IaC est un élément essentiel pour automatiser le provisionnement de l'infrastructure, s'intégrant parfaitement dans notre pipeline DevSecOps.

# PROBLÉMATIQUE

La transition vers des pratiques DevSecOps agiles et sécurisées présente des défis significatifs pour les organisations modernes.

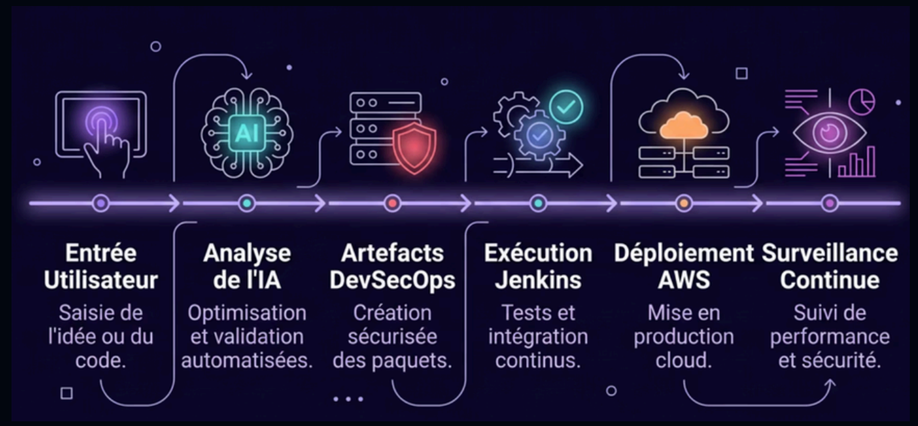
## Défis Majeurs

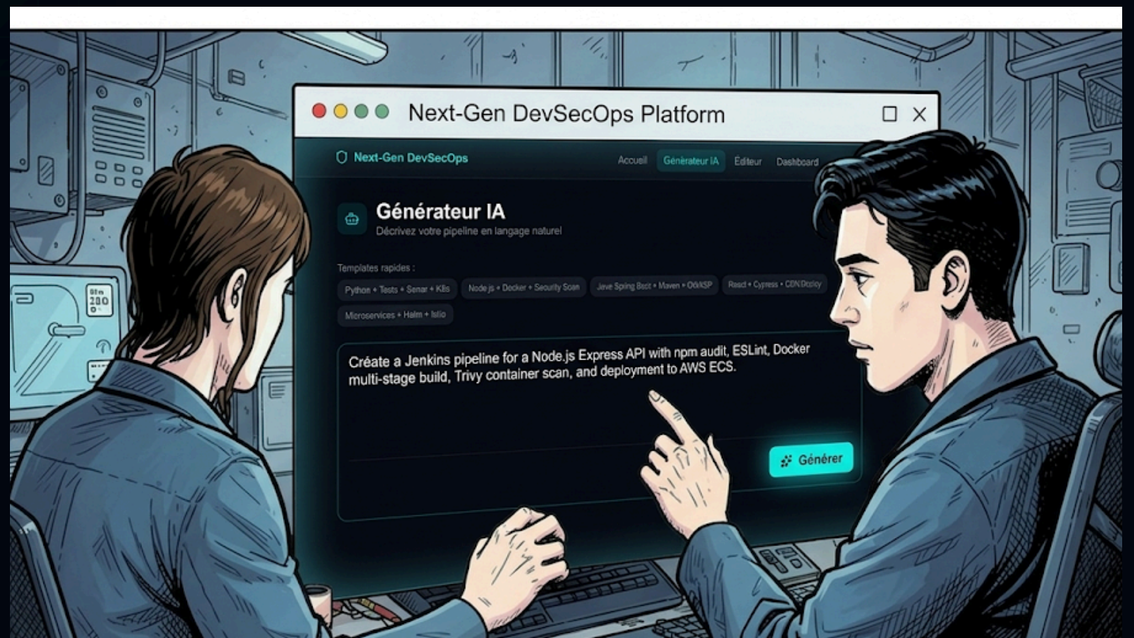
- ·Configurations manuelles complexes : Sources d'erreurs et de lenteur.
- ·Outils multiples et déconnectés : Fragmentation des workflows.
- ·Sécurité tardive (Shift-Right) : Vulnérabilités détectées trop tard, coûts élevés.
- ·Complexité du cloud : Gestion ardue des environnements et ressources.
- ·Intégration CI/CD difficile : Peu d'automatisation de bout en bout.
- ·Risques accrus : Vulnérabilités et erreurs de configuration fréquentes.

# « COMMENT AUTOMATISER ET SÉCURISER UN WORKFLOW DEVSECOPS COMPLET À PARTIR D'UNE SIMPLE REQUÊTE EN LANGAGE NATUREL ? »

# NOTRE SOLUTION

Une plateforme intelligente qui transforme les requêtes en langage naturel en workflows DevSecOps sécurisés et automatisés.

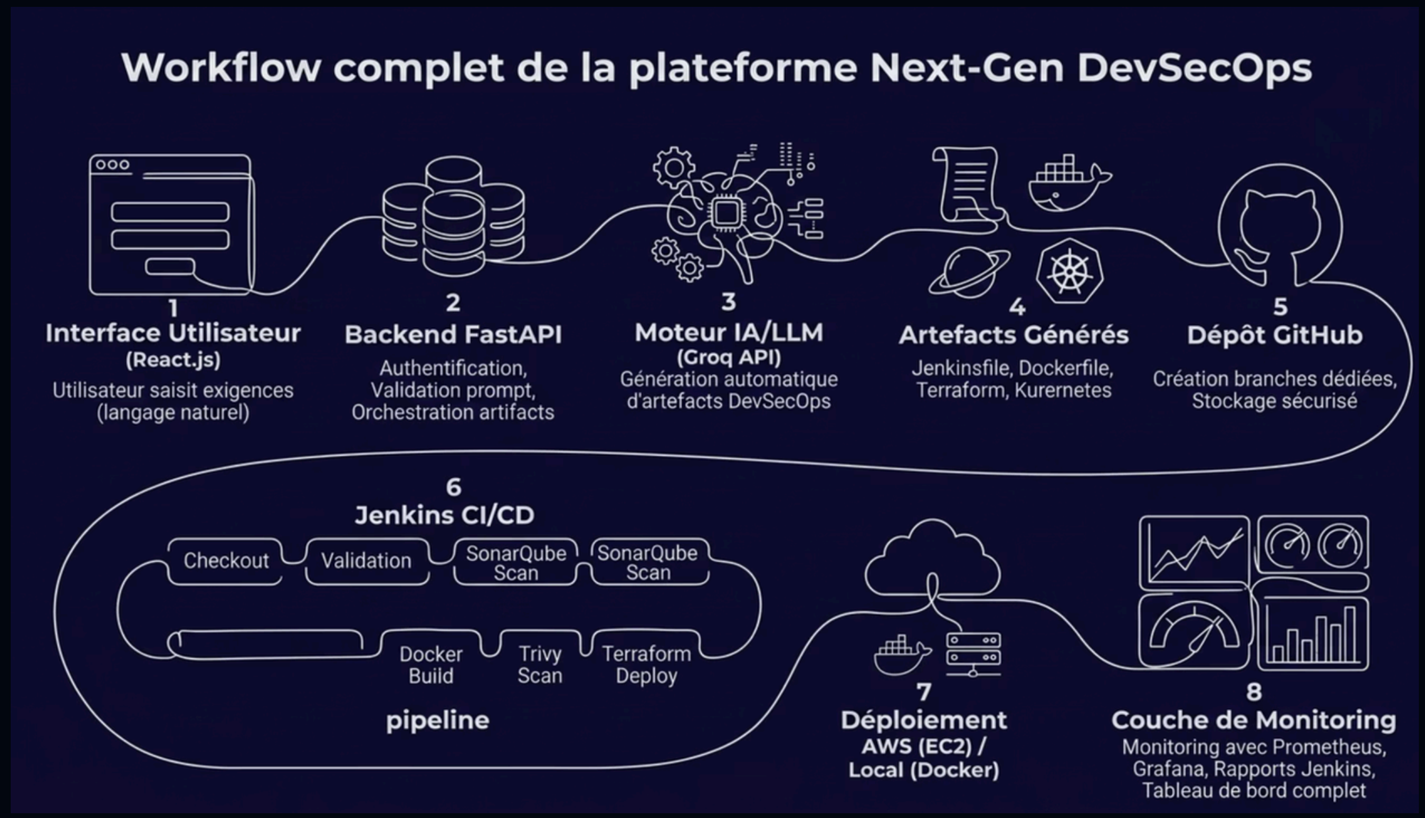



# WORKFLOW DU PROJET

LE PROJET NEXT-GEN DEVSECOPS PROPOSE UNE PLATEFORME INTELLIGENTE POUR AUTOMATISER LE CYCLE DEVSECOPS. FACE AUX CONFIGURATIONS MANUELLES ET AUX INTÉGRATIONS DE SÉCURITÉ TARDIVES, NOTRE SOLUTION TRANSFORME UNE DESCRIPTION EN LANGAGE NATUREL EN ARTEFACTS DEVSECOPS EXPLOITABLES.

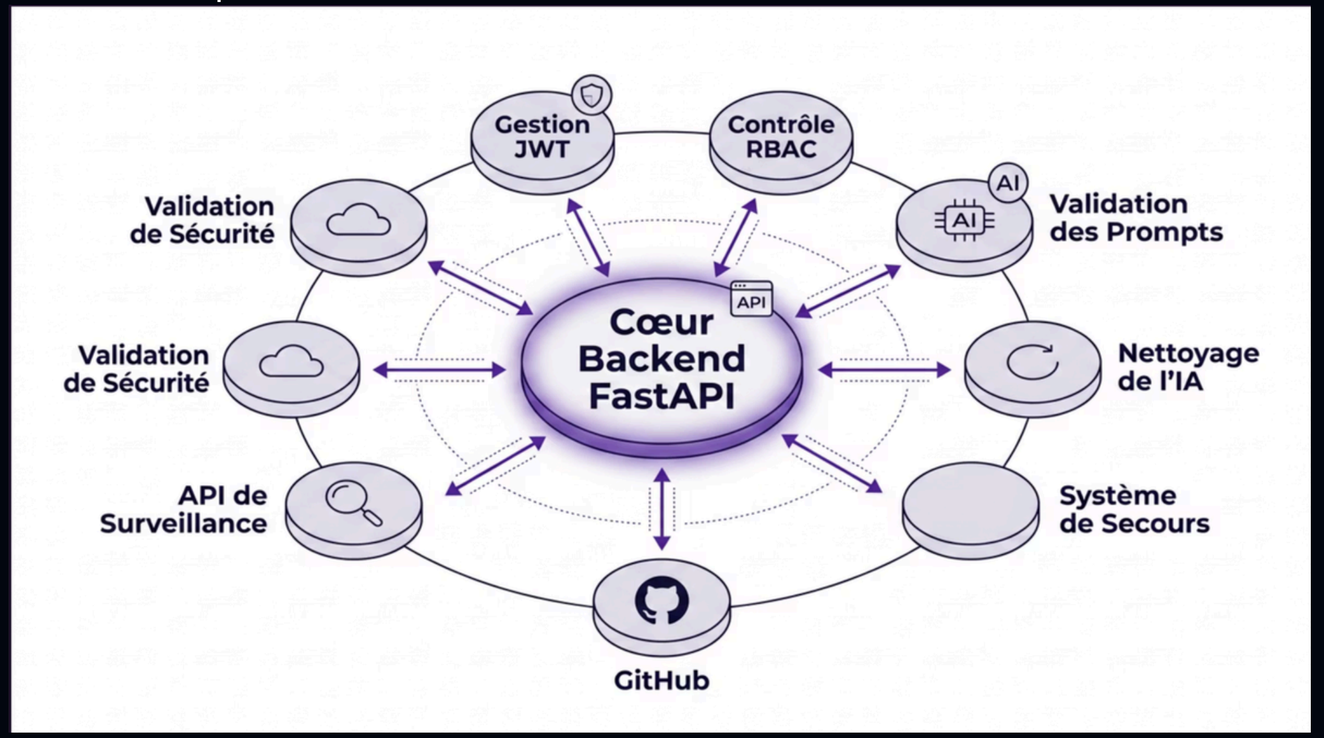
## FONCTIONNEMENT




- Interface web (React.js) pour la saisie des besoins.
- Backend (FastAPI) sécurise le prompt et utilise une API LLM.
- Génération de Jenkinsfile, Terraform, Dockerfile, manifeste Kubernetes.
- Normalisation, validation et poussée vers GitHub.
- Jenkins exécute le pipeline (SonarQube, Trivy, déploiement AWS EC2).
- Monitoring et rapports d'exécution (Prometheus, Grafana, frontend).

# BACKEND : CERVEAU DU SYSTÈME

Le backend est le moteur intelligent de la plateforme, orchestrant chaque interaction et garantissant un flux DevSecOps fluide et sécurisé.




**« TRANSFORME LA SORTIE DE L'IA EN WORKFLOWS EXÉCUTABLES, VALIDÉS ET SÉCURISÉS. »**

# ARTEFACTS GÉNÉRÉS

Notre plateforme automatise la création des composants essentiels pour un déploiement rapide et sécurisé.

## ARTEFACTS GÉNÉRÉS AUTOMATIQUEMENT :

### FICHIERS TERRAFORM

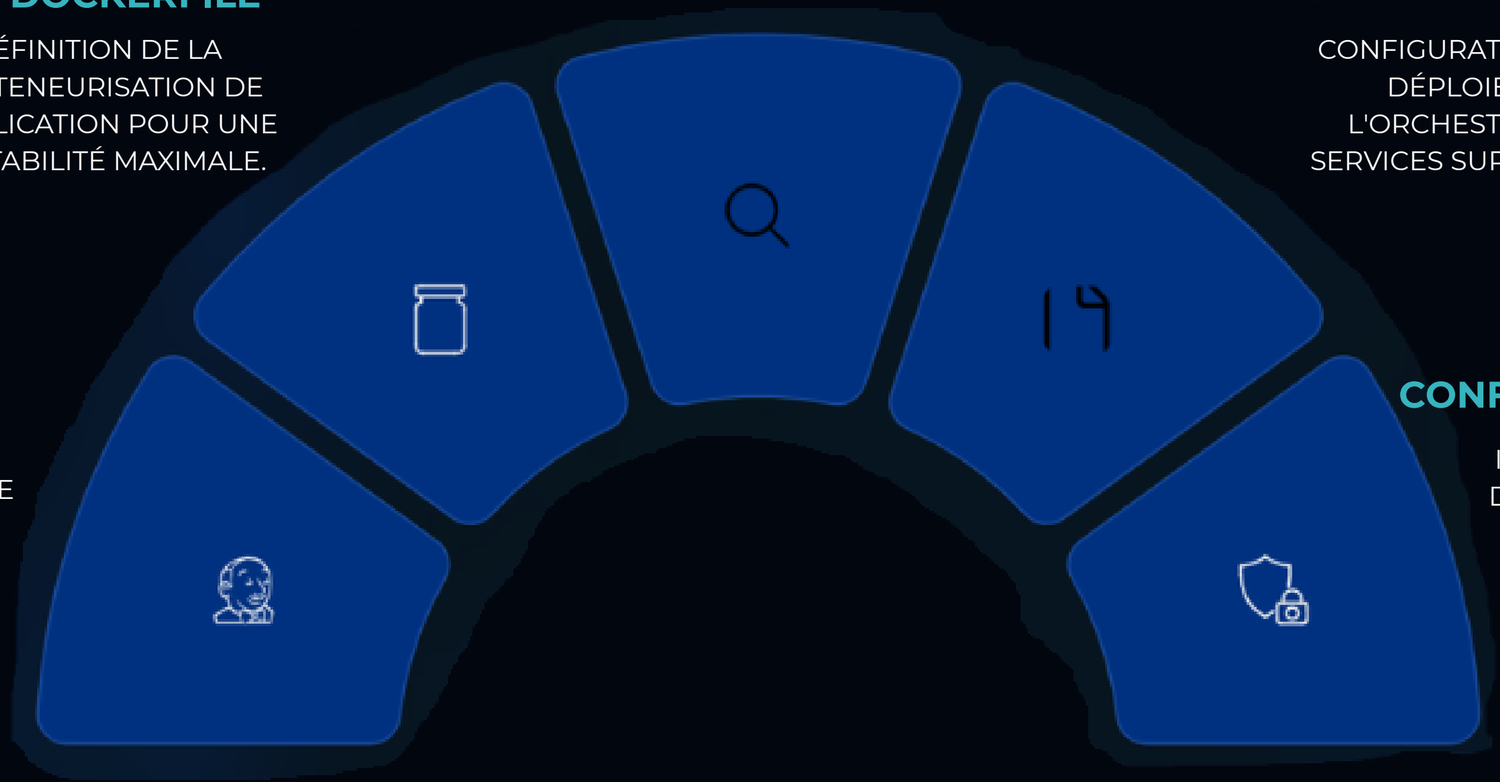
CODE D'INFRASTRUCTURE AS CODE (IAC) POUR UN PROVISIONNEMENT AUTOMATISÉ ET REPRODUCTIBLE.

### DOCKERFILE

DÉFINITION DE LA CONTENEURISATION DE L'APPLICATION POUR UNE PORTABILITÉ MAXIMALE.

### FICHIERS KUBERNETES

CONFIGURATIONS POUR LE DÉPLOIEMENT ET L'ORCHESTRATION DES SERVICES SUR KUBERNETES.



### JENKINSFILE

SCRIPT ORCHESTRANT LES ÉTAPES DU PIPELINE CI/CD, DE L'INTÉGRATION AU DÉPLOIEMENT.

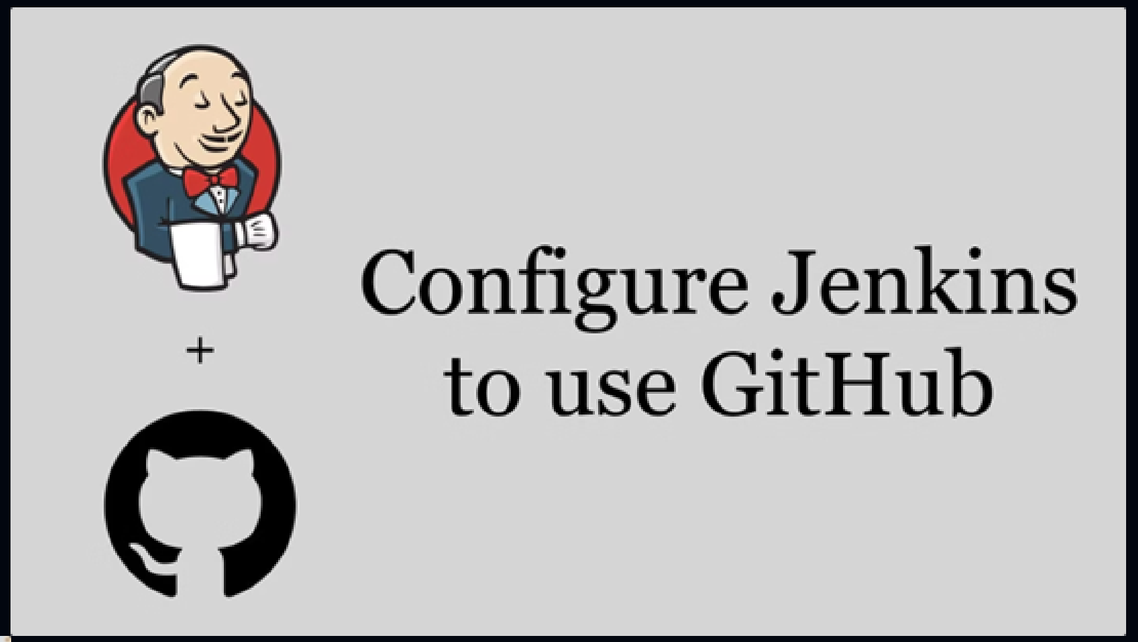
### CONFIGURATIONS SÉCURITÉ

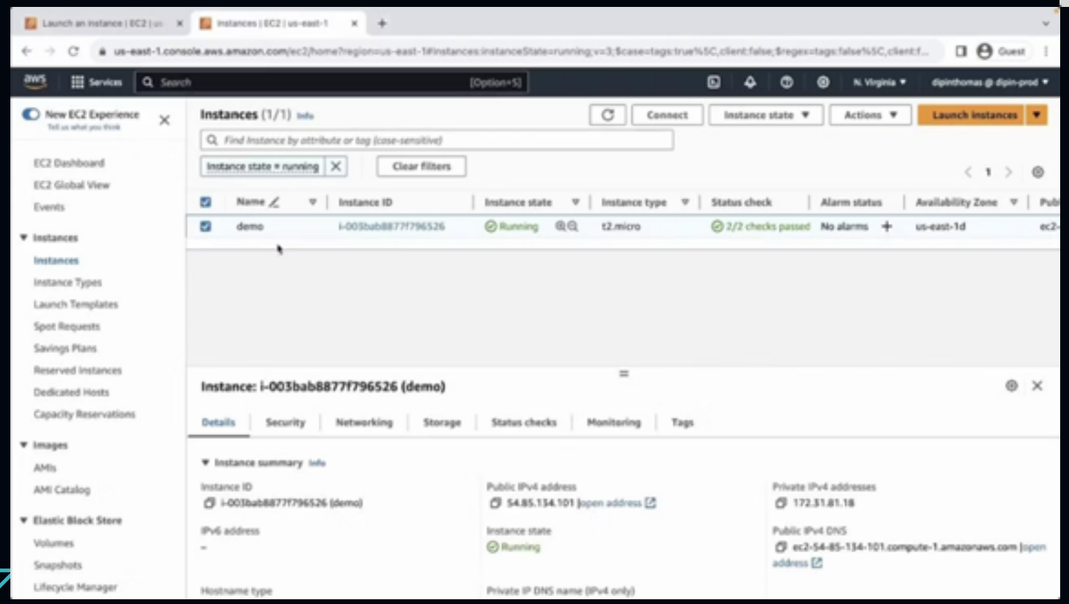
INTÉGRATION DES POLITIQUES DE SÉCURITÉ ET DES OUTILS DE MONITORING DÈS LA GÉNÉRATION.

# INTÉGRATION DE JENKINS ET AWS

## Connexion GitHub

- **Dépôt GitHub dédié pour les pipelines générés.**
- **Credentials Jenkins sécurisés pour l'accès au dépôt privé.**
- **Multibranch Pipeline pour la détection automatique des branches.**




## Déploiement AWS

- •Utilisateur AWS dédié avec permissions limitées.
- •Credentials AWS ajoutés dans Jenkins.
- •Stage Terraform Deploy pour créer l'instance EC2.
- •Récupération de l'IP publique pour l'accès à l'application.

# SÉCURITÉ CÔTÉ APPLICATION ET IA

LA SÉCURITÉ COMMENCE DÈS L'ENTRÉE UTILISATEUR, AVANT MÊME LA GÉNÉRATION DES ARTEFACTS.

**1-** LOGIN SÉCURISÉ
**2-** AUTHENTIFICATION PAR JWT
**3-** RBAC SELON LES RÔLES UTILISATEUR
**4-** PROMPT GUARD CONTRE LES PROMPTS DANGEREUX
**5-** BLOCAGE DES DEMANDES SENSIBLES
**6-** VALIDATION DES SORTIES GÉNÉRÉEAS PAR L'IA
**7 -** GÉNÉRATION D'ARTEFACTS SÉCURISÉS

# PIPELINE CI/CD JENKINS

LE PIPELINE GÉNÉRÉ PERMET D'EXÉCUTER AUTOMATIQUEMENT LES ÉTAPES DEVSECOPS APRÈS LA GÉNÉRATION DES ARTEFACTS.

| Declarative: Checkout SCM | Checkout | Validate | SonarQube Analysis | Docker Build & Push | Security Scan | Terraform Deploy | Declarative: Post Actions |
|---|---|---|---|---|---|---|---|
| 1s | 798ms | 967ms | 14s | 9s | 28s | 1min 8s | 891ms |
| 1s | 798ms | 967ms | 14s | 9s | 28s | 1min 8s | 891ms |

# PIPELINE CI/CD JENKINS

1- CHECKOUT DU CODE DEPUIS GITHUB.

2- VALIDATION DES ARTEFACTS GÉNÉRÉS.

3- ANALYSE STATIQUE AVEC SONARQUBE.

4- BUILD DE L'IMAGE DOCKER.

5- SCAN DE VULNÉRABILITÉS AVEC TRIVY.

6- DÉPLOIEMENT AUTOMATIQUE AVEC TERRAFORM.

7- GÉNÉRATION ET ENVOI DU RAPPORT JENKINS.

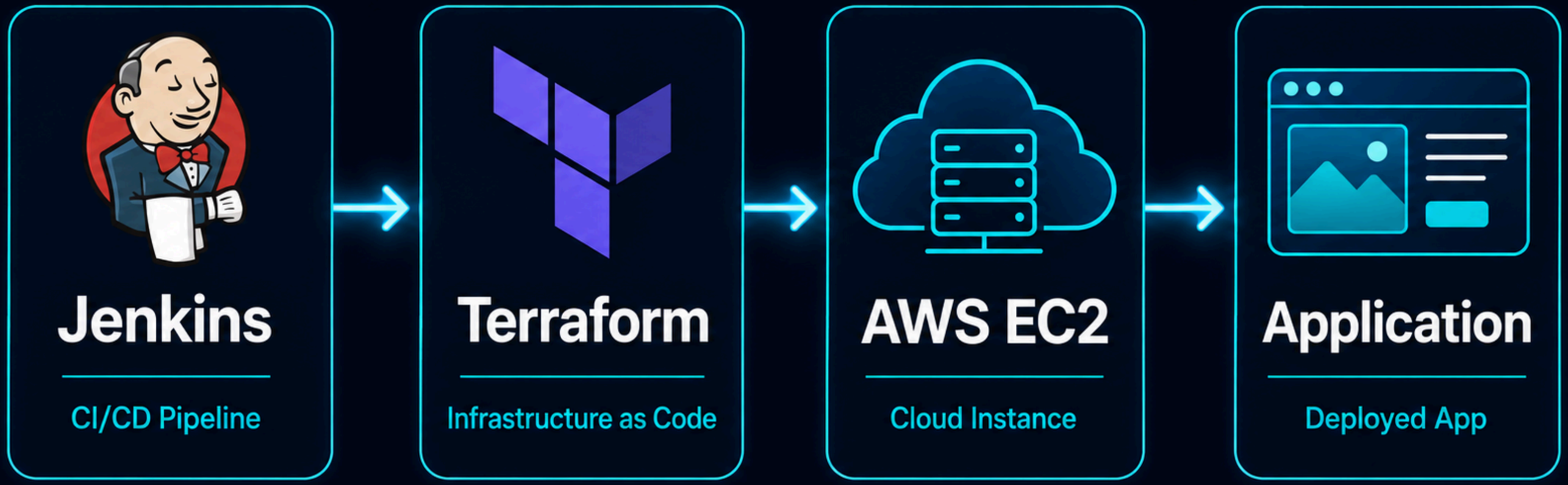
# POURQUOI MITRE ATT&CK ?

MITRE ATT&CK PERMET DE RELIER LES RISQUES TECHNIQUES À DES TECHNIQUES D'ATTAQUE RÉELLES.

1- RÉFÉRENTIEL RECONNU EN CYBERSÉCURITÉ
2- CLASSIFICATION DES TECHNIQUES D'ATTAQUE
3- MEILLEURE COMPRÉHENSION DES RISQUES
4- AIDE AU SCORING SÉCURITÉ
5- EXPLICATION CLAIRE DES ALERTES
6- LIEN ENTRE IA, DEVSECOPS ET SÉCURITÉ

# DÉPLOIEMENT AWS

TERRAFORM PERMET DE CRÉER AUTOMATIQUEMENT L'INFRASTRUCTURE AWS.



# DÉPLOIEMENT LOCAL RAPIDE

LE TEST LOCAL PERMET DE VALIDER LES ARTEFACTS AVANT LE CLOUD.

**1-** Build de l’image Docker
**2-** Lancement du conteneur localement
**3-** Test rapide de l’application
**4-** Validation avant AWS
**5-** Réduction des erreurs
**6-** Économie des ressources cloud

# MONITORING ET OBSERVABILITÉ

Le monitoring permet de suivre l’état de la plateforme et des pipelines.

### Métriques & Visualisation

**Prometheus** assure la collecte continue des métriques systèmes et applicatives, tandis que **Grafana** centralise et affiche l'ensemble des dashboards dynamiques.

### Observabilité API (FastAPI)

Analyse en temps réel du trafic grâce au **suivi des requêtes HTTP**, couplé à une surveillance précise de **la latence** globale de réponse.

### Supervision des Pipelines CI/CD

Connexion directe pour **le suivi des builds Jenkins** permettant **une visualisation claire des succès et des échecs** de chaque intégration.

# La page Monitoring.

## Tableau de bord des pipelines

| Total des rapports Jenkins | Builds réussis | Builds échoués |
|---|---|---|
| 4 | 2 | 2 |

| Taux de succès | Durée moyenne d'exécution | Niveau de risque dominant |
|---|---|---|
| 50% | 42.2s | LOW |

### Résultats des pipelines


### Déploiements cette semaine


### Répartition des risques


### Rapports Jenkins

Mis à jour après chaque pipeline

# RAPPORT JENKINS
# DANS LE FRONTEND

**Génération du Rapport après Pipeline**

À l'issue de chaque exécution, **Jenkins génère un rapport complet** capturant les détails essentiels : **le statut, la durée de build** et **la branche déployée**.

**Scans de Sécurité et Audit Qualité**

Restitution instantanée des analyses de vulnérabilités d'image via **Trivy** et de la qualité du code statique via **SonarQube.**

**Transmission Backend et Monitoring**

**Envoi fluide des métriques vers le backend** pour alimenter l'interface utilisateur avec **l'URL de déploiement** finale au sein de la page Monitoring.

# DIFFICULTÉS RENCONTRÉES

### Intégration Jenkins avec Docker

Difficulté de configuration liée aux **montages de sockets** et à la gestion des permissions au sein d'environnements conteneurisés d'intégration.

### Sorties IA parfois invalides

Gestion des anomalies de génération avec mise en place de **mécanismes de validation d'artefacts** robustes.

### Configuration Prometheus / Jenkins

Complexité du ciblage et du scraping des endpoints de métriques, nécessitant **un paramétrage réseau robuste.**

### Gestion des credentials sensibles

Sécurisation rigoureuse des variables d'environnement et de l'accès aux APIs distantes (Jenkins, AWS, Docker Hub).

### Nettoyage des fichiers inutiles

Contraintes de maintenance liées à l'accumulation **d'images Docker orphelines** et d'artefacts temporaires volumineux.

### Risque de coûts AWS

Nécessité d'optimiser l'orchestration des pipelines de tests automatisés **pour limiter la consommation de ressources cloud.**

# Améliorations futures

Feuille de route stratégique et technique pour les prochaines étapes de développement.

PHASE 1 - COURT TERME

### IA & Robustesse

- → Assistant IA conversationnel intégré
- → Auto-correction des artefacts générés

PHASE 2 - MOYEN TERME

### Simulation & Coûts

- → Simulation AWS complète avec LocalStack
- → Estimation en temps réel du coût AWS

PHASE 3 - LONG TERME

### Résilience & Sec

- → Mécanisme de rollback automatique
- → Mise en place d'un scoring DevSecOps global

PHASE 4 - VISION

### Échelle & Cloud

- → Stratégie de déploiement multi-cloud agile

# DEMO

# CONCLUSION

**Next-Gen DevSecOps combine l'IA, le CI/CD, l'Infrastructure as Code, la sécurité, le cloud et le monitoring dans une seule plateforme.**

- Automatisation intelligente
- Sécurité intégrée par défaut
- Déploiement hybride local & cloud
- Monitoring centralisé 360°
- Réduction drastique des erreurs humaines