Projet DevOps : Automatisation du Déploiement d'une Application Web sur AWS

NACERE Mohammed

27 octobre 2025

Table des matières

1 Objectif du projet

Le projet **DevOps-project** a pour objectif de mettre en place une chaîne d'intégration et de déploiement continu (CI/CD) permettant d'automatiser le **build, le test et le déploiement** d'une application web Java sur une instance **AWS EC2**.

Ce projet illustre la démarche DevOps, en combinant les aspects **développement** (build Maven) et **exploitation** (automatisation du déploiement, gestion des services et scripts d'installation).

2 Architecture technique

L'architecture du projet se compose des éléments suivants :

— Application Java Web développée avec **JSP/Servlets** et packagée via **Maven** (`pom.xml`),
— Déploiement sur un serveur **EC2** sous Ubuntu,
— Automatisation à l'aide de **scripts Bash**,
— Intégration et déploiement continus définis dans les fichiers `buildspec.yml` et `appspec.yml`, utilisés dans la chaîne **AWS CodeBuild / CodeDeploy**.

Schéma conceptuel simplifié

```
Développeur → Commit GitHub → CodeBuild (buildspec.yml)
             → Artifact (ZIP) → CodeDeploy (appspec.yml)
             → EC2 Instance (scripts de déploiement)
```

3 Fonctionnement du pipeline CI/CD

3.1 Étape d'intégration continue (CI)

Le fichier `buildspec.yml` décrit les étapes d'intégration :

— Installation des dépendances Maven,
— Compilation du projet (`mvn package`),
— Génération de l'artefact `.war`,
— Archivage des fichiers nécessaires au déploiement (code, scripts, configuration).

Ce fichier est exécuté automatiquement par **AWS CodeBuild** à chaque commit sur le dépôt GitHub.

3.2 Étape de déploiement continu (CD)

Le fichier `appspec.yml` est utilisé par **AWS CodeDeploy**. Il définit les actions suivantes :

— Copie des fichiers sur l'instance EC2,
— Appel des scripts de déploiement :
  — **BeforeInstall** : arrêt du service (`stop_server.sh`),
  — **AfterInstall** : installation des dépendances (`install_dependencies.sh`),
  — **ApplicationStart** : démarrage du serveur (`start_server.sh`).

Ces étapes garantissent un **déploiement automatisé, reproductible et sans interruption majeure**.

4 Description des scripts Bash

4.1 `install_dependencies.sh`

Ce script prépare l'environnement avant le lancement de l'application :

```
#!/bin/bash
sudo apt update -y
sudo apt install -y java-17-amazon-corretto-devel
```

Objectif : garantir que le serveur dispose de toutes les dépendances nécessaires.

4.2 `stop_server.sh`

Ce script arrête proprement le service avant déploiement :

```
#!/bin/bash
sudo systemctl stop tomcat || true
```

Objectif : éviter les conflits ou fichiers verrouillés lors de la mise à jour.

4.3 `start_server.sh`

Ce script relance le service applicatif à la fin du déploiement :

```
#!/bin/bash
sudo systemctl start tomcat
```

Objectif : s'assurer que la nouvelle version de l'application est bien en ligne.

5 Partie applicative

La partie applicative est composée d'une **page JSP** (`index.jsp`) et d'un fichier `web.xml` dans `WEB-INF/`. Ces fichiers représentent la structure d'une application Java EE classique déployée sur Tomcat.

6 Points techniques clés

Aspect	Technologies / Outils	Description
Build	Maven	Compilation et empaquetage du projet
CI/CD	AWS CodeBuild / CodeDeploy	Automatisation du build et du déploiement
Cloud	AWS EC2	Serveur hébergeant l'application
Scripting	Bash	Automatisation des installations et redémarrages
Déploiement	appspec.yml / buildspec.yml	Définition des étapes du pipeline
Système	Linux (Ubuntu)	Environnement serveur cible

7 Bénéfices et apports personnels

Ce projet m'a permis de :

— Comprendre le cycle complet de vie d'une application (développement, build, déploiement),
— Mettre en œuvre les principes de **CI/CD** et d'automatisation,
— Approfondir mes compétences en **Linux**, **scripting Bash** et **serveurs**,
— Découvrir l'environnement **Cloud AWS** et la logique **Infrastructure as Code**,
— Expérimenter les **bonnes pratiques DevOps** : versioning, déploiement reproductible, traçabilité.

8 Conclusion

Ce projet illustre la mise en œuvre d'une approche DevOps complète, réunissant intégration, automatisation et déploiement continu autour d'une application Java.

L'utilisation de scripts Bash et de services AWS démontre comment industrialiser le déploiement tout en maintenant la fiabilité et la traçabilité des livraisons applicatives.