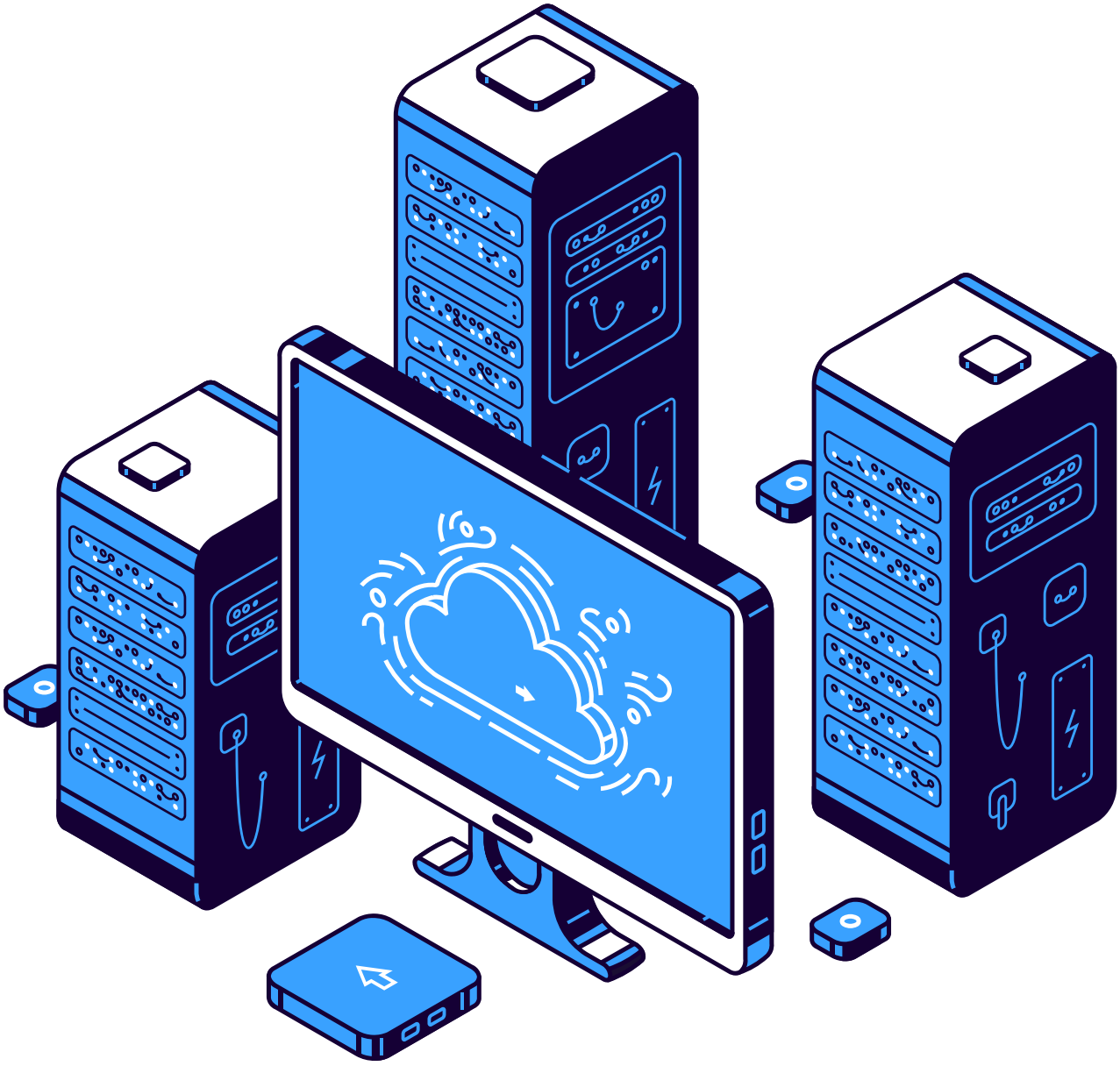
RAPPORT DEVOPS

Preparer par :
Nabil Abdelwahd et Sabir Abdessamd



Objectif

Déployer une application web statique de votre choix dans différents environnements de déploiement (simulés sur AWS).

Pipeline CI/CD sur Jenkins

1. **Build** : Cloner le projet et construire une image Docker de l'application avec ses dépendances et un serveur Nginx.

```
environment {
    AWS_SSH_KEY = credentials('aws-key.pem')
    DOCKER_IMAGE = "yassine112/mon-app-web"
    VERSION = "${env.BUILD_NUMBER ?: 'latest'}"
    REVIEW_IP = "51.21.180.149"
    STAGING_IP = "51.20.56.9"
    PROD_IP = "13.60.156.76"
}

stages {
    stage('Checkout Code') {
        steps {
            git branch: 'main',
                credentialsId: 'github-credentials',
                url: 'https://github.com/YassineDev32/Tp_Jenkins_Docker_AWS.git'
        }
    }
```

2. **Test** : Vérifier le bon fonctionnement et l'accessibilité de l'image Docker avant son déploiement. Exécuter l'image en local et tester la disponibilité avec `curl`.

```
stage('Test Image') {
    steps {
        script {
            powershell '''
                try {
                    # Verify image exists locally
                    $imageExists = docker images -q "${env:DOCKER_IMAGE}:${env:VERSION}"
                    if (-not $imageExists) {
                        throw "Image ${env:DOCKER_IMAGE}:${env:VERSION} doesn't exist locally"
                    }

                    # Run container
                    docker run -d -p 8081:80 --name test-container "${env:DOCKER_IMAGE}:${env:VERSION}"
                    Start-Sleep -Seconds 10

                    # Test application
                    $response = Invoke-WebRequest -Uri "http://localhost:8081" -UseBasicParsing -ErrorActi
                    if ($response.StatusCode -ne 200) {
                        throw "HTTP Status ${response.StatusCode}"
                    }
                    Write-Host "Test passed successfully"
                } catch {
                    Write-Host "Test failed: $_"
                    docker logs test-container
                    exit 1
                } finally {
                    # Cleanup container
                    docker stop test-container -t 1 | Out-Null
                    docker rm test-container -f | Out-Null
                }
            '''
        }
    }
}
```

3. **Release** : Pousser l'image sur Docker Hub.

```
stage('Login to Docker Hub') {
    steps {
        script {
            withCredentials([
                usernamePassword(
                    credentialsId: 'docker-hub-creds',
                    usernameVariable: 'DOCKER_USER',
                    passwordVariable: 'DOCKER_PASS'
                )
            ]) {
                powershell '''
                    # Clear existing credentials
                    docker logout
                    Remove-Item -Path "$env:USERPROFILE/.docker/config.json" -Force -ErrorAction SilentlyContinue

                    # Create Docker config directory if it doesn't exist
                    $dockerConfigDir = "$env:USERPROFILE/.docker"
                    if (-not (Test-Path $dockerConfigDir)) {
                        New-Item -ItemType Directory -Path $dockerConfigDir -Force | Out-Null
                    }

                    # Create auth token (base64 encoded username:password)
                    $authToken = [Convert]::ToBase64String(
                        [Text.Encoding]::ASCII.GetBytes("${env:DOCKER_USER}:${env:DOCKER_PASS}")
                    )

                    # Create the Docker config file without here-string issues
                    $dockerConfigContent = '{
                        "auths": {
                            "https://index.docker.io/v1/": {
                                "auth": "' + $authToken + '"
                            }
                        }
                    }'

                    $dockerConfigContent | Out-File -FilePath "$dockerConfigDir/config.json" -Encoding ascii

                    # Verify the login works
                    docker pull hello-world
                    if ($LASTEXITCODE -ne 0) {
                        throw "Docker authentication verification failed"
                    }

                    Write-Host "Successfully authenticated with Docker Hub"
                '''
            }
        }
    }
}
```

```
stage('Push to Docker Hub') {
    steps {
        script {
            powershell '''
                # Push with retries
                $maxRetries = 3
                $retryCount = 0
                do {
                    docker push "${env:DOCKER_IMAGE}:${env:VERSION}"
                    if ($LASTEXITCODE -eq 0) {
                        Write-Host "Successfully pushed ${env:DOCKER_IMAGE}:${env:VERSION}"
                        break
                    }
                    $retryCount++
                    if ($retryCount -lt $maxRetries) {
                        Start-Sleep -Seconds 10
                        Write-Host "Retrying push ($retryCount/$maxRetries)..."
                    }
                } while ($retryCount -lt $maxRetries)

                if ($retryCount -eq $maxRetries) {
                    throw "Failed to push after $maxRetries attempts"
                }
            '''
        }
    }
}
```

1. **Deploy in Review** : Exécuter l'image poussée sur Docker Hub pour la déployer sur une machine AWS Ubuntu, par exemple.

```
stage('Deploy to Review') {
    steps {
        withCredentials([file(credentialsId: 'aws-key.pem', variable: 'SSH_KEY')]) {
            script {
                // 1. Prepare the key file with proper permissions
                powershell '''
                    $tempKey = "$env:TEMP\\aws-key-$env:BUILD_NUMBER.pem"

                    # Copy key content with Unix line endings
                    [System.IO.File]::WriteAllText(
                        $tempKey,
                        [System.IO.File]::ReadAllText($env:SSH_KEY).Replace("`r`n","`n"),
                        [System.Text.Encoding]::ASCII
                    )

                    # Set strict permissions
                    icacls $tempKey /inheritance:r
                    icacls $tempKey /grant:r "$env:USERNAME:(R)"
                    icacls $tempKey /grant:r "SYSTEM:(R)"
                '''

                // 2. Execute deployment commands with proper waiting
                powershell '''
                    $sshCommand = "docker pull ${env:DOCKER_IMAGE}:${env:VERSION} && " +
                                  "docker stop review-app || true && " +
                                  "docker rm review-app || true && " +
                                  "docker run -d -p 80:80 --name review-app ${env:DOCKER_IMAGE}:${env:VERSION}"

                    $process = Start-Process -FilePath "ssh" `
                        -ArgumentList @(
                            "-i", "$env:TEMP\\aws-key-$env:BUILD_NUMBER.pem",
                            "-o", "StrictHostKeyChecking=no",
                            "ubuntu@${env:REVIEW_IP}",
                            $sshCommand
                        ) `
                        -NoNewWindow `
                        -PassThru `
                        -Wait

                    if ($process.ExitCode -ne 0) {
                        throw "SSH command failed with exit code $($process.ExitCode)"
                    }
                '''

                // 3. Clean up
                powershell '''
                    Remove-Item "$env:TEMP\\aws-key-$env:BUILD_NUMBER.pem" -Force -ErrorAction SilentlyContinue
                '''
            }
        }
    }
}
```

2. **Deploy in Staging** : Répéter le processus dans un environnement de préproduction (staging).

```
stage('Deploy to Staging') {
    steps {
        withCredentials([file(credentialsId: 'aws-key.pem', variable: 'SSH_KEY')]) {
            script {
                powershell '''
                    $tempKey = "$env:TEMP\\aws-key-staging-$env:BUILD_NUMBER.pem"

                    # Create key file
                    [System.IO.File]::WriteAllText(
                        $tempKey,
                        [System.IO.File]::ReadAllText($env:SSH_KEY).Replace("`r`n","`n"),
                        [System.Text.Encoding]::ASCII
                    )

                    # Set permissions
                    icacls $tempKey /inheritance:r
                    icacls $tempKey /grant:r "$env:USERNAME:(R)"
                    icacls $tempKey /grant:r "SYSTEM:(R)"

                    # Deployment commands
                    $commands = @(
                        "docker pull ${env:DOCKER_IMAGE}:${env:VERSION}",
                        "docker stop staging-app || true",
                        "docker rm staging-app || true",
                        "docker run -d -p 80:80 --name staging-app ${env:DOCKER_IMAGE}:${env:VERSION}"
                    ) -join " && "

                    # Execute with retries
                    $maxRetries = 3
                    $retryCount = 0
                    do {
                        try {
                            $process = Start-Process ssh -ArgumentList @(
                                "-i", $tempKey,
                                "-o", "StrictHostKeyChecking=no",
                                "-o", "ConnectTimeout=30",
                                "ubuntu@${env:STAGING_IP}",
                                $commands
                            ) -NoNewWindow -PassThru -Wait

                            if ($process.ExitCode -ne 0) {
                                throw "SSH failed with exit code $($process.ExitCode)"
                            }
                            break
                        } catch {
                            $retryCount++
                            if ($retryCount -ge $maxRetries) {
                                throw
                            }
                            Start-Sleep -Seconds 10
                            Write-Host "Retrying deployment ($retryCount/$maxRetries)..."
                        }
                    } while ($true)

                    # Cleanup
                    Remove-Item $tempKey -Force -ErrorAction SilentlyContinue
                '''
            }
        }
    }
}
```

3. **Deploy in Production** : Simuler un déploiement final en production.

```
stage('Deploy to Production') {
    steps {
        withCredentials([file(credentialsId: 'aws-key.pem', variable: 'SSH_KEY')]) {
            script {
                powershell '''
                    $tempKey = "$env:TEMP\\aws-key-prod-$env:BUILD_NUMBER.pem"

                    # Create key file
                    [System.IO.File]::WriteAllText(
                        $tempKey,
                        [System.IO.File]::ReadAllText($env:SSH_KEY).Replace("`r`n","`n"),
                        [System.Text.Encoding]::ASCII
                    )

                    # Set permissions
                    icacls $tempKey /inheritance:r
                    icacls $tempKey /grant:r "$env:USERNAME:(R)"
                    icacls $tempKey /grant:r "SYSTEM:(R)"

                    # Deployment commands
                    $commands = @(
                        "docker pull ${env:DOCKER_IMAGE}:${env:VERSION}",
                        "docker stop prod-app || true",
                        "docker rm prod-app || true",
                        "docker run -d -p 80:80 --name prod-app ${env:DOCKER_IMAGE}:${env:VERSION}"
                    ) -join " && "

                    # Execute with retries
                    $maxRetries = 3
                    $retryCount = 0
                    do {
                        try {
                            $process = Start-Process ssh -ArgumentList @(
                                "-i", $tempKey,
                                "-o", "StrictHostKeyChecking=no",
                                "-o", "ConnectTimeout=30",
                                "ubuntu@${env:PROD_IP}",
                                $commands
                            ) -NoNewWindow -PassThru -Wait

                            if ($process.ExitCode -ne 0) {
                                throw "SSH failed with exit code $($process.ExitCode)"
                            }
                            break
                        } catch {
                            $retryCount++
                            if ($retryCount -ge $maxRetries) {
                                throw
                            }
                            Start-Sleep -Seconds 10
                            Write-Host "Retrying deployment ($retryCount/$maxRetries)..."
                        }
                    } while ($true)

                    # Cleanup
                    Remove-Item $tempKey -Force -ErrorAction SilentlyContinue
                '''
            }
        }
    }
}
```


```
Last login: Thu Mar 27 16:53:08 2025 from 105.155.130.36
ubuntu@ip-172-31-43-7:~$ docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED         STATUS         PORTS
               NAMES
7bc562f5242c   yassine112/mon-app-web:126     "/docker-entrypoint.…"   3 minutes ago   Up 3 minutes   0.0.0.0:80->80/tcp,
:::80->80/tcp   prod-app
ubuntu@ip-172-31-43-7:~$
```

```
[Pipeline] { (Deploy to Review)
[Pipeline] withCredentials
Masking supported pattern matches of %SSH_KEY%
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] powershell
fichier trait��: C:\WINDOWS\TEMP\aws-key-126.pem
1 fichiers correctement trait�s�; �chec du traitement de 0 fichiers
powershell.exe : Param�tre non valide ��(R)��
Au caractère C:\ProgramData\Jenkins\.jenkins\workspace\Déployer une application web statique AWS@tmp\durable-60c7d29c\powershellWrapper.ps1:3
: 1
+ & powershell -NoProfile -NonInteractive -ExecutionPolicy Bypass -Comm ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (Param�tre non valide ��(R)��:String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

fichier trait��: C:\WINDOWS\TEMP\aws-key-126.pem
1 fichiers correctement trait�s�; �chec du traitement de 0 fichiers
[Pipeline] powershell
126: Pulling from yassine112/mon-app-web
Digest: sha256:2c7c72e217996e41a3414dd71162b151c881ba6d2403a86eefdee164a973a01a
Status: Downloaded newer image for yassine112/mon-app-web:126
docker.io/yassine112/mon-app-web:126
review-app
review-app
dfed06417777b16d821fbabf67154dd28a299c37319b6b664f34c187cea1d5d26
[Pipeline] powershell
```

```
[Pipeline] { (Deploy to Staging)
[Pipeline] withCredentials
Masking supported pattern matches of %SSH_KEY%
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] powershell
fichier trait��: C:\WINDOWS\TEMP\aws-key-staging-126.pem
1 fichiers correctement trait�s�; �chec du traitement de 0 fichiers
powershell.exe : Param�tre non valide ��(R)��
Au caractère C:\ProgramData\Jenkins\.jenkins\workspace\Déployer une application web statique AWS@tmp\durable-74e8b678\powershellWrapper.ps1:3
: 1
+ & powershell -NoProfile -NonInteractive -ExecutionPolicy Bypass -Comm ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (Param�tre non valide ��(R)��:String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

fichier trait��: C:\WINDOWS\TEMP\aws-key-staging-126.pem
1 fichiers correctement trait�s�; �chec du traitement de 0 fichiers
126: Pulling from yassine112/mon-app-web
6e909acdb790: Pulling fs layer
5eaa34f5b9c2: Pulling fs layer
417c4bccf534: Pulling fs layer
e7e0ca015e55: Pulling fs layer
373fe654e984: Pulling fs layer
97f5c0f51d43: Pulling fs layer
c22eb46e871a: Pulling fs layer
74a18cf642ea: Pulling fs layer
```

```
97f5c0f51d43: Waiting
c22eb46e871a: Waiting
74a18cf642ea: Waiting
e7e0ca015e55: Waiting
417c4bccf534: Verifying Checksum
417c4bccf534: Download complete
6e909acdb790: Download complete
e7e0ca015e55: Verifying Checksum
e7e0ca015e55: Download complete
5eaa34f5b9c2: Verifying Checksum
5eaa34f5b9c2: Download complete
373fe654e984: Verifying Checksum
373fe654e984: Download complete
97f5c0f51d43: Verifying Checksum
97f5c0f51d43: Download complete
c22eb46e871a: Verifying Checksum
c22eb46e871a: Download complete
6e909acdb790: Pull complete
74a18cf642ea: Download complete
5eaa34f5b9c2: Pull complete
417c4bccf534: Pull complete
e7e0ca015e55: Pull complete
373fe654e984: Pull complete
97f5c0f51d43: Pull complete
c22eb46e871a: Pull complete
74a18cf642ea: Pull complete
Digest: sha256:2c7c72e217996e41a3414dd71162b151c881ba6d2403a86eefdee164a973a01a
Status: Downloaded newer image for yassine112/mon-app-web:126
docker.io/yassine112/mon-app-web:126
```

```
74a18cf642ea: Download complete
5eaa34f5b9c2: Pull complete
417c4bccf534: Pull complete
e7e0ca015e55: Pull complete
373fe654e984: Pull complete
97f5c0f51d43: Pull complete
c22eb46e871a: Pull complete
74a18cf642ea: Pull complete
Digest: sha256:2c7c72e217996e41a3414dd71162b151c881ba6d2403a86eefdee164a973a01a
Status: Downloaded newer image for yassine112/mon-app-web:126
docker.io/yassine112/mon-app-web:126
Error response from daemon: No such container: staging-app
Error response from daemon: No such container: staging-app
43cef1ea3429be017233d960e3e8a1faf3b068fd067393eb7faa1c36d5f8d3b9
[Pipeline] }
[Pipeline] // script
```

```
[Pipeline] { (Deploy to Production)
[Pipeline] withCredentials
Masking supported pattern matches of %SSH_KEY%
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] powershell
fichier trait��: C:\WINDOWS\TEMP\aws-key-prod-126.pem
1 fichiers correctement trait�s�; �chec du traitement de 0 fichiers
powershell.exe : Param�tre non valide ��(R)��
Au caractère C:\ProgramData\Jenkins\.jenkins\workspace\Déployer une application web statique AWS@tmp\durable-67f0c9cc\powershellWrapper.ps1:3
: 1
+ & powershell -NoProfile -NonInteractive -ExecutionPolicy Bypass -Comm ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (Param�tre non valide ��(R)��:String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

fichier trait��: C:\WINDOWS\TEMP\aws-key-prod-126.pem
1 fichiers correctement trait�s�; �chec du traitement de 0 fichiers
Warning: Permanently added '13.60.156.76' (ED25519) to the list of known hosts.
126: Pulling from yassine112/mon-app-web
6e909acdb790: Pulling fs layer
5eaa34f5b9c2: Pulling fs layer
417c4bccf534: Pulling fs layer
e7e0ca015e55: Pulling fs layer
373fe654e984: Pulling fs layer
97f5c0f51d43: Pulling fs layer
c22eb46e871a: Pulling fs layer
```

```
c22eb46e871a: Verifying Checksum
c22eb46e871a: Download complete
74a18cf642ea: Verifying Checksum
74a18cf642ea: Download complete
6e909acdb790: Pull complete
5eaa34f5b9c2: Pull complete
417c4bccf534: Pull complete
e7e0ca015e55: Pull complete
373fe654e984: Pull complete
97f5c0f51d43: Pull complete
c22eb46e871a: Pull complete
74a18cf642ea: Pull complete
Digest: sha256:2c7c72e217996e41a3414dd71162b151c881ba6d2403a86eefdee164a973a01a
Status: Downloaded newer image for yassine112/mon-app-web:126
docker.io/yassine112/mon-app-web:126
Error response from daemon: No such container: prod-app
Error response from daemon: No such container: prod-app
7bc562f5242cdc3915a47f9773acc2826f9463fbefd84667b562b1213bce4e61
[Pipeline] }
```

Déployer une application web statique AWS

Liens permanents

- Dernier build (#126), il y a 10 mn
- Dernier build stable (#126), il y a 10 mn
- Dernier build avec succès (#126), il y a 10 mn
- Dernier build en échec (#125), il y a 16 mn
- Dernier build non réussi (#125), il y a 16 mn
- Dernier build complété (#126), il y a 10 mn