|  |
| --- |
| onepoint |
| AWS DevOps Competency |
| Case Study |

|  |
| --- |
|  |

# **Case Study Requirements:**

# Each case study must be specific to the AWS Competency applying for.

# **About the Customer**

GRTgaz is a major operator in the high-pressure gas transportation sector. The company has a public service mission aimed at guaranteeing the continuity of gas transmission, and a genuine commitment to promoting renewable gas and to the energy transition in the territories. (Source : <https://www.grtgaz.com/en/our-group/grtgaz-in-brief>)

# **Customer Challenge**

En 2018, GRTgaz initie avec onepoint la Fabrique Technologies Dynamiques afin de répondre au développement de son activité et au souhait d’améliorer la qualité de son système d’information et d’optimiser la gestion de ses services informatiques. Les objectifs sont divers : industrialiser le build des solutions applicatives, améliorer la performance et la qualité du delivery, mise en œuvre de méthodologies adaptées aux contextes et enjeux, réduire le coût global, etc.

La volonté du client est de raccourcir les couts de maintenance et Time To Market en replaçant le métier au centre des décisions. GRTgaz initie alors des groupements d’applications et afin de répondre aux objectifs, le groupement NTI traité par ce dossier concerne 8 applications qui ont été migrées dans le Cloud.

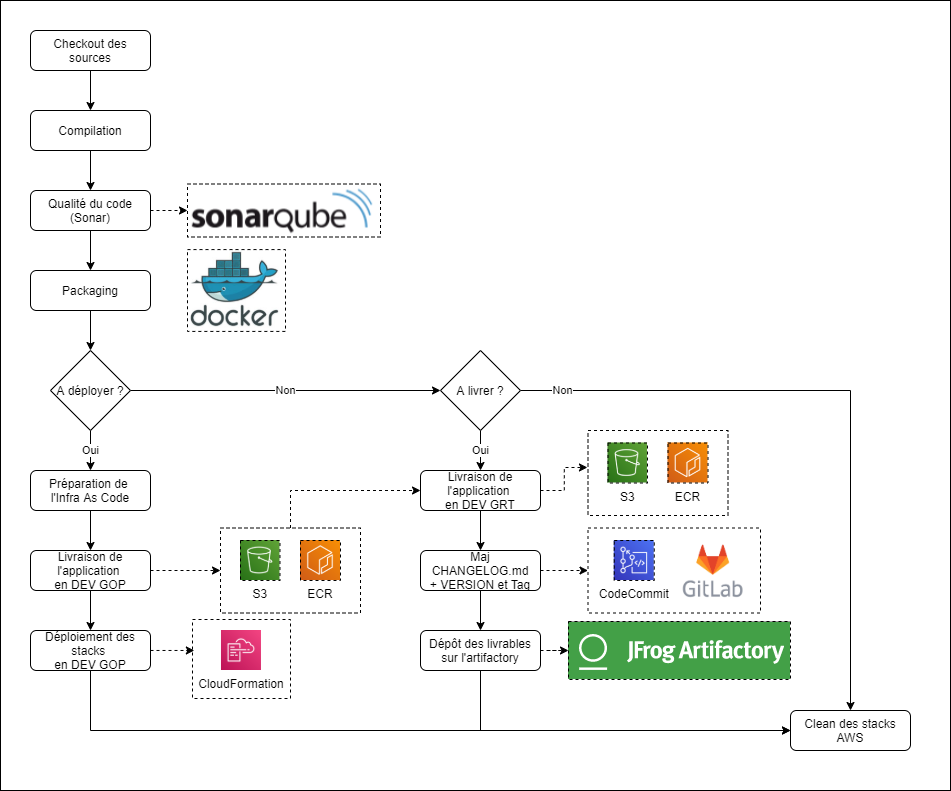
TODO : Ajouter le fait que toute cette mécanique était en infogérance extérieure (donc couteux et pas réactif)

# **Partner Solution**

Les 8 applications du groupement NTI ont été migrées dans le cloud AWS, celle-ci étant basées sur de nombreuses technologies plus ou moins récentes : JAVA, .NET, Silverlight, etc. Un certain nombre de choix techniques ont été effectués lors de cette bascule :

* Passage de Oracle à PostgreSQL
* Changement du serveur d’application
* Dockerisation des applications
* Intégration à AWS (Cognito,Cloudwatch,etc)

Ces applications sont hébergées sur Gitlab. La CICD gérée par Jenkins via Jenkinsfile. Le déploiement s’effectue ensuite sur un environnement d’intégration AWS avec sceptre et CloudFormation, il est possible de déployer plusieurs environnements en parallèle via un système de feature deployment basé sur le nom des branches git.

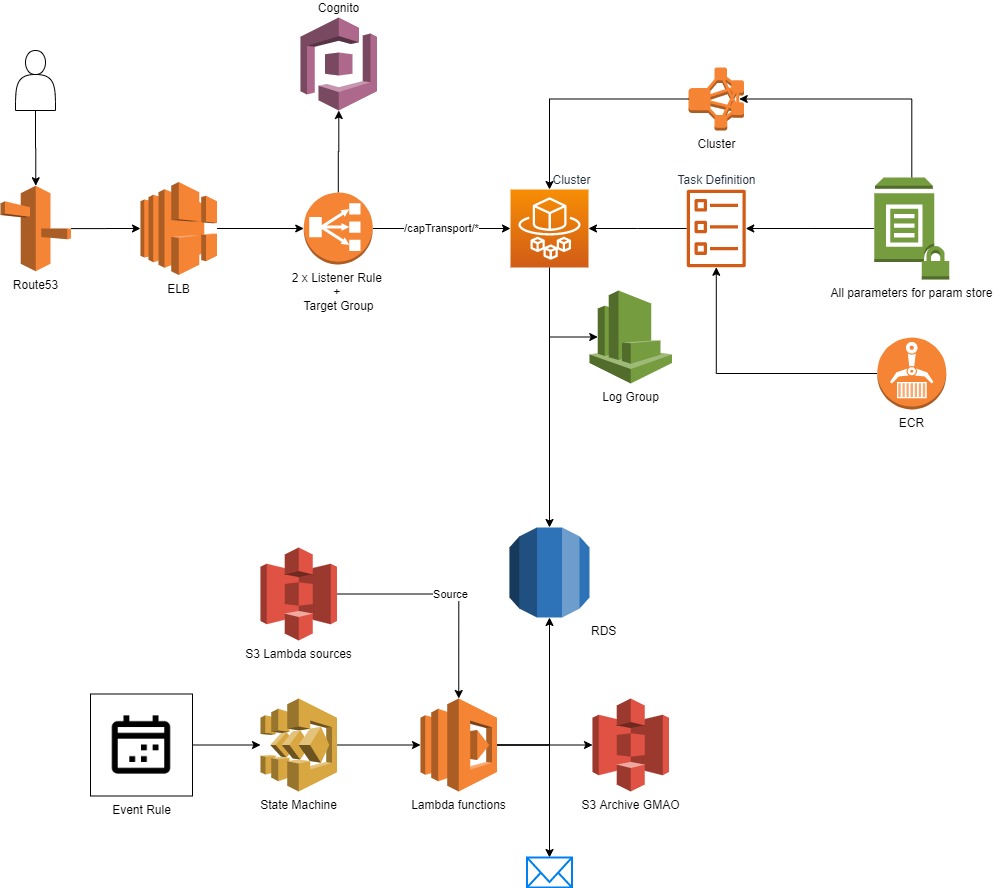


La livraison d’une application s’effectue par dépôt de nos livrables sur des briques AWS : S3/ECR/CodeCommit. Cette livraison est réceptionnée par le Cloud Center of Excellence de GRTgaz qui gère le déploiement des applications sur des environnement DEV/STG/PROD côté GRTgaz, chaque environnement déployé sur un compte AWS distinct.

NOTE : Peut être mettre qu’on est admin en DEV GRT et read only.

L’infrastructure de nos applications est gérée en As Code avec sceptre. Un projet d’infrastructure parent permet d’abord d’établir la configuration des éléments réseau (VPC, Subnets, Security Group, Rôle & Profile, etc.) ainsi que d’établir des templates sceptre pour les applications. L’infrastructure des applications est ensuite découpée dans plusieurs fichiers avec des références aux templates du projet d’infrastructure parent.





D’autres projets nous ont amené à mettre en place d’autres briques AWS : API Gateway, Batch.

NOTE : Contextualiser API Gateway, Batch et lambdas.

En complément de sceptre, nous avons aussi scripté certaines configurations AWS de nos environnements via des script SHELL en s’appuyant sur AWS CLI (ajout d’users de tests dans cognito, configuration d’users BDD, enregistrement de callbacks cognito pour le feature deployment, etc).

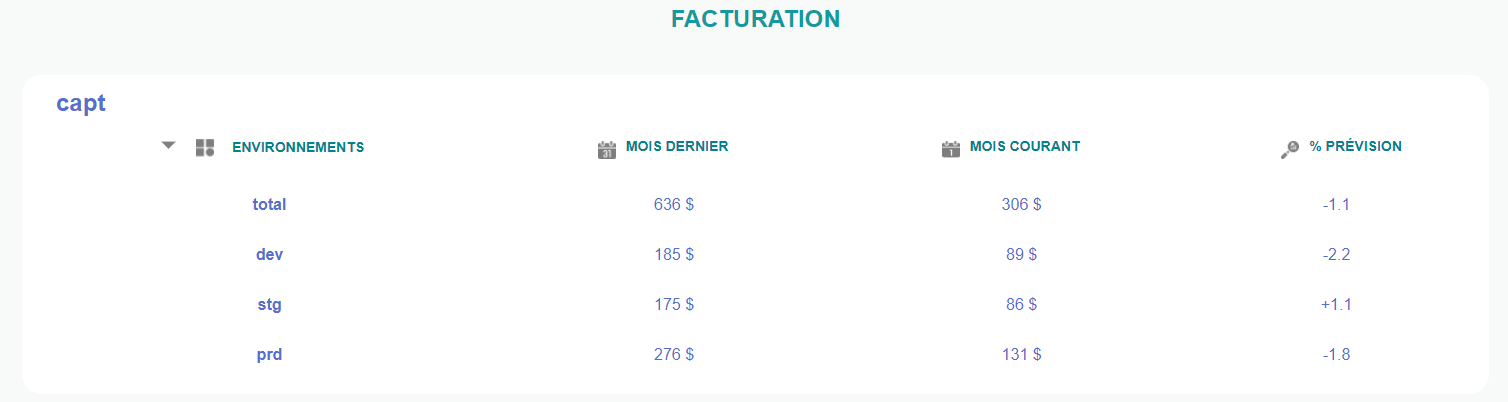
# **Results and Benefits**

# 

La migration Cloud des applications NTI a permis de raccourcir le temps de mise en production d’une évolution ou correction qui peut s’effectuer en moins de 15 minutes. De même, les déploiements applicatifs peuvent être opérés directement par des utilisateurs métier qualifiés.

En termes de coût, la tarification de chaque environnement peut être suivie finement et est prévisible d’un mois sur l’autre. Les environnements DEV/STG GRT sont généralement éteint et leur démarrage passe par une action utilisateur. Sur notre environnement AWS onepoint, les applications sont par défaut éteintes si on ne travaille pas dessus.

Exemple de suivi des coûts :



# NOTE : Mettre une partie sur les évolutions à venir, passage terraform, usine logicielle, etc

# **About the Partner** -

* A remplir par onepoint