SOMMAIRE

RESUME EXECUTIF

L'architecture technique est une vue tournée sur l'organisation logique de la plateforme informatique, c'est-à-dire les moyens techniques clés qui seront utilisés par tous les logiciels applicatifs. La vue contient le matériel informatique, les logiciels systèmes, les middlewares, ainsi que les réseaux de télécommunication et les relations entre ces différents éléments. Pour une entreprise ou une institution, le choix de l'architecture technique vise à maximiser les possibilités d'implantation de logiciels du commerce ainsi que de réalisation de logiciels sur mesure. Il vise également à rentabiliser l'utilisation du matériel et des logiciels déjà acquis par l'institution. Pour une entreprise ou une institution qui transforme son architecture technique, le plan d'architecture est accompagné d'un planning et d'un budget des acquisitions, des ventes et des opérations de migration nécessaires pour aligner le système informatique avec le plan. L'architecture logicielle est une vue tournée sur l'organisation interne et le découpage d'un logiciel en modules.

INTRODUCTION (Contexte, hypothèses, exigences)

Le marché tchadien des jeux vidéo est de plus en plus croissant de marques telles que Sony, Microsoft importées pour l'essentiel de Dubaï (Emirats Arabes Unis) qui constituent l'écosystème des jeux vidéo importés. La société de jeux souhaite faire la promotion de ses services en ligne via une application sécurisée.

A cet effet, elle sollicite notre expertise.

Compte tenu des enjeux, nous présentons quelques hypothèses suivantes :

- Une solution sécurisée conçue avec le Framework Angular et avec un langage informatique tel que JavaScript.
- Une solution sécurisée conçue avec le Framework Symfony et avec un langage informatique tel que PHP.

Le contenu de notre mémoire comporte plusieurs parties à savoir l'Architecture cible, ensuite les Détails de conception, ensuite la Sécurité (IAM, chiffrement, réseau, conformité), ensuite le Déploiement (IaC, CI/CD, environnements, rollback), ensuite les Observabilité & opérations (dashboards, alertes, runbooks), ensuite les Tests & résultats (méthode, outils, mesures, limites), ensuite les Coûts & optimisation (FinOps), ensuite les Risques, les limites, les perspectives et enfin les Références & annexes.

A- Architecture cible :

A travers le développement d'une application Front end avec le Framework Angular.

Le cahier de charges

- **Présentation de la société**

Présentation de l'activité de la société : L'activité principale de la société «GUEGUE IT» est de vendre des consoles de Playstation 4, de XBOX ONE et des CD de jeux vidéo. Expression des besoins : Besoin de réalisation d'une petite application afin de l'utiliser pendant la période de promotion fixé pour les vacances d'été entre mai et août. Explication du rôle de l'application dans la stratégie de la société : Le rôle de l'application dans la stratégie de la société est de passer l'information sur les prix des CD de jeux vidéo pendant la période de promotion. Conception : Conception de la solution et identification des risques. Indication de la date butoir pour la fin du projet : Mise en ligne de l'application escomptée après le 18/05/2025.

- **Cible à laquelle s'adresse l'application**

Cible : La cible toute personne souhaitant acheter des CD de jeux vidéo moins chères. Lieu de la cible : De manière générale, la cible se trouve sur internet, à travers des canaux numériques qu'elle fréquente tels que les réseaux sociaux.

- **Aspects d'ergonomie et de graphisme**

Couleurs principales de l'application : Les couleurs principales de l'application sont le vert et le blanc. Utilisation du logo : Le logo utilisé est celui de la société, au niveau du côté gauche de l'entête. Police de caractère mise en place sur l'application : La police de caractère mise en place sur l'application est diverse et variée. Design ou effets particuliers : Pour le cas du design ou des effets particuliers, nous utiliserons HTML 5 pour le contenu, CSS 3 pour la présentation, Bootstrap 5 pour rendre l'application responsive et TypeScript (JavaScript) pour amener des animations sur l'application.

- **Aspect fonctionnel de l'application**

L'application requiert une page web où le visiteur peut visualiser les différents CD de jeux vidéo de PS4 et XBOX ONE qui sont à vendre. Au niveau de la présentation de chaque CD, nous avons une petite fonction d'incrémentation et de décrémentation au niveau du nombre afin de voir le coût, sachant également que tous

ces CD ont le même prix. Et au niveau du pied de page, nous avons le contact du vendeur.

- **Aspects techniques de l'application**

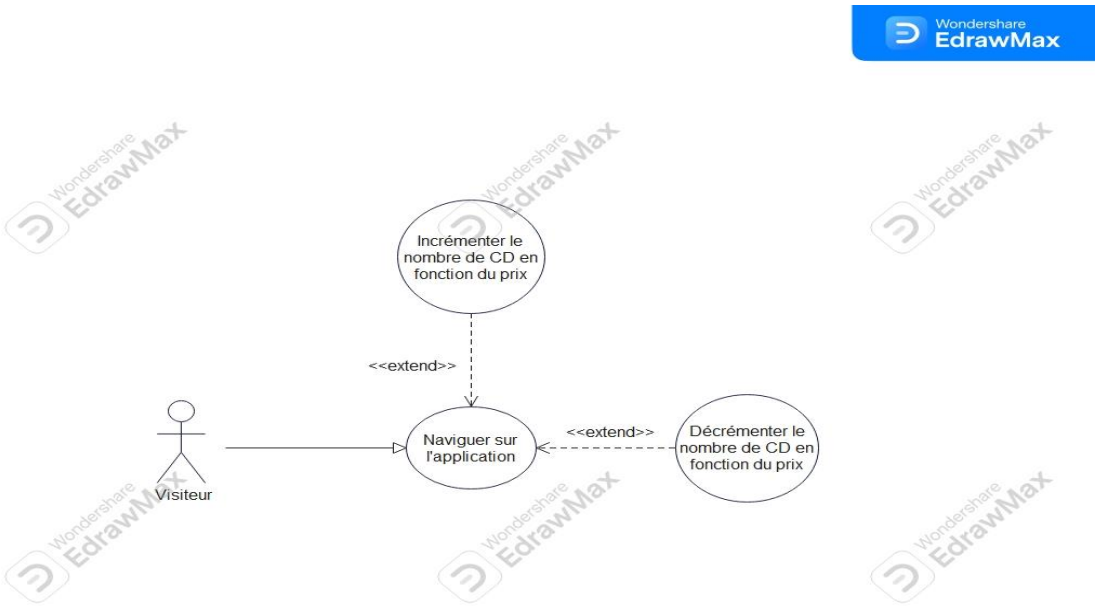
Solution technique : La solution technique que nous privilégions pour le développement de l'application est l'utilisation du Framework Angular de Google à travers le logiciel open source Visual Studio Code qui est un environnement de développement intégré ou IDE. Cas du nom de domaine et de l'hébergement : Le nom de domaine et l'hébergement s'obtiendra avec Firebase de Google qui est un ensemble d'outils pour l'hébergement et le développement d'applications mobiles et web.

B- <u>Détails de conception</u> :

Modélisation UML

UML (**U**nified **M**odeling **L**anguage ou Langage de modélisation unifié) est un langage de modélisation fondé sur le concept objet. L'objectif d'UML est de fournir une notation standard utilisable dans le développement des systèmes informatiques basés sur l'objet.

- **Diagramme de cas d'utilisation Visiteur**



<u>Fiche de Cockburn user case</u>

Fiche de Cockburn user case

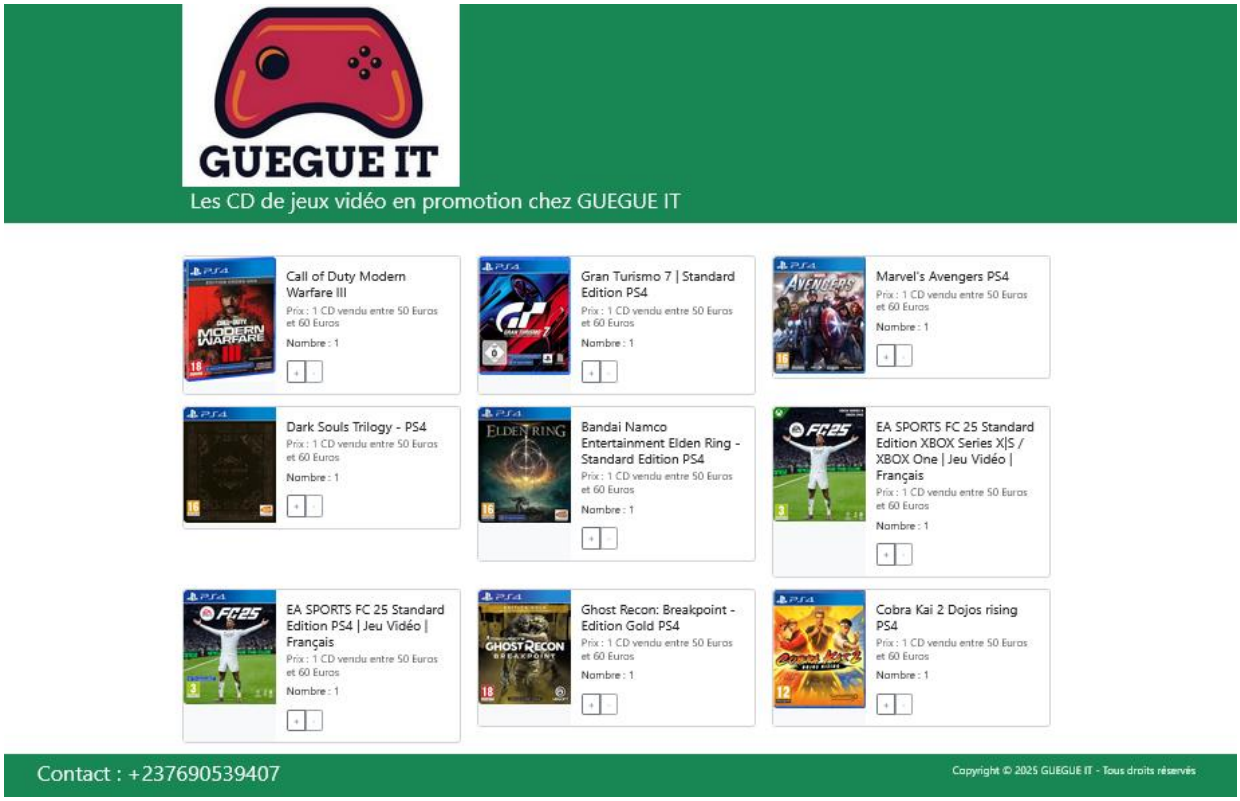
- Titre : Diagramme de cas d'utilisation Visiteur.
- Résumé : Nous avons la représentation du visiteur qui peut naviguer sur
- l'application, en faisant l'incrémentation et la décrémentation du nombre de CD en fonction du prix.
- Acteur : Visiteur.
- Motivation : L'acteur veut naviguer sur l'application.
- Pré-condition(s) : Aucune.
- Post-condition(s) : Aucune.
- Exceptions : Aucune.
- Remarques ergonomiques (éventuellement) : Aucune.
- Contraintes non fonctionnelles (éventuellement) : Aucune.
- Scénario Nominal : [Enchaînement de Cockburn].

Sécurité dans Angular

Angular intègre de manière native des processus avancés de protection contre les attaques les plus répandues connues sous le nom de Cross-Site Scripting (XSS), ainsi que des attaques sous les noms d'injection SQL et de Cross Site Request Forgery (CSRF).

Présentation de l'application

Notre application est disponible en ligne à travers le lien suivant : https://angularguegueit-app.web.app/



C- Sécurité (IAM, chiffrement, réseau, conformité) :

Politique associée : dev.json

Dev User a les permissions suivantes : Upload / Download de fichiers ; Consultation que de son bucket, créer un bucket supplémentaire ; Il a le droit de supprimer les buckets qu'il a créé ; Ils ne voient pas ceux des autres (ceux qu'il n'a pas créer) et Il utilise S3 / EC2 (droit ec2).

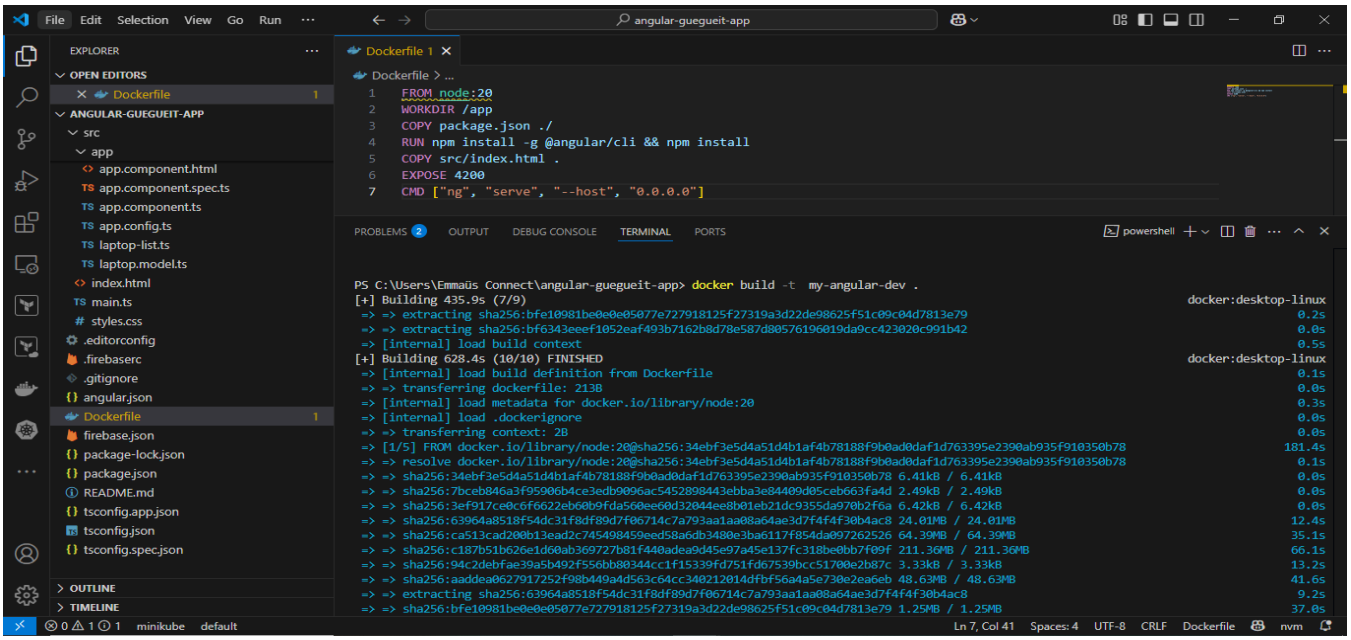
Nous avons le code json disponible sur le referentiel du code en archive .zip

D- Déploiement (IaC, CI/CD, environnements, rollback) :

Création de nœuds avec Minikube

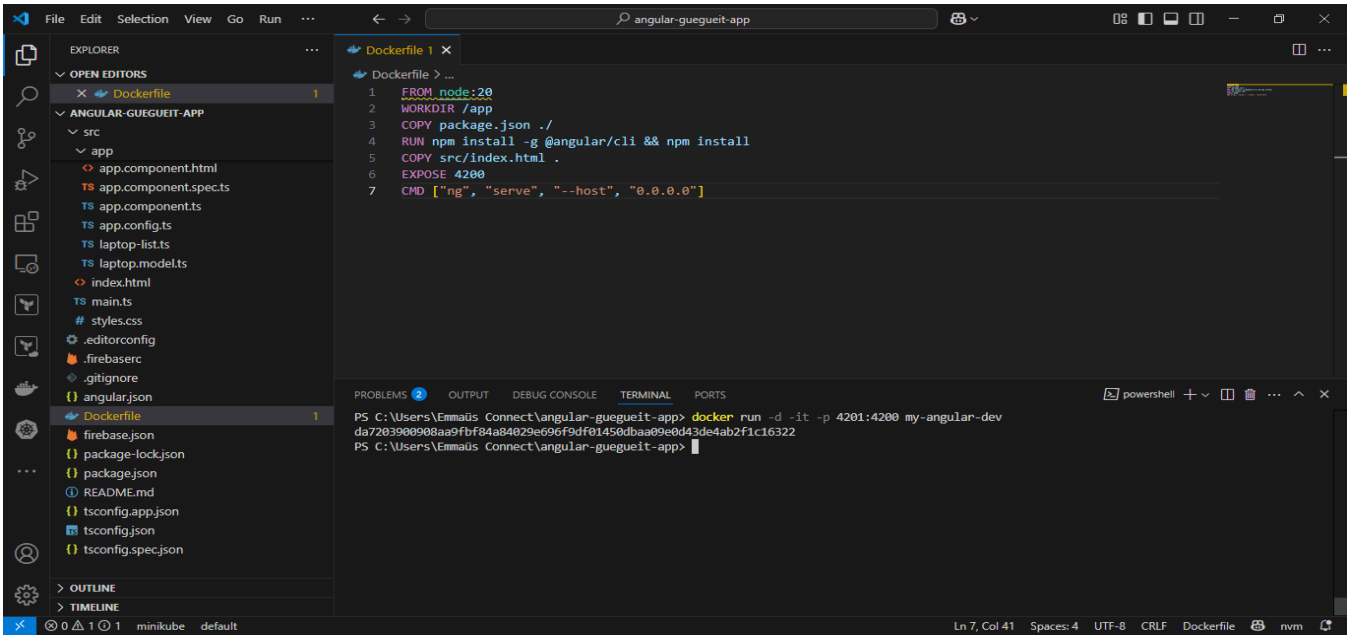
- **Etape 1 :**

Nous allons tout d'abord créer le Dockerfile et taper la commande "docker build –t my-angular-dev ." qui nous permet de créer une image Docker à partir du Dockerfile situé dans le répertoire courant (.).

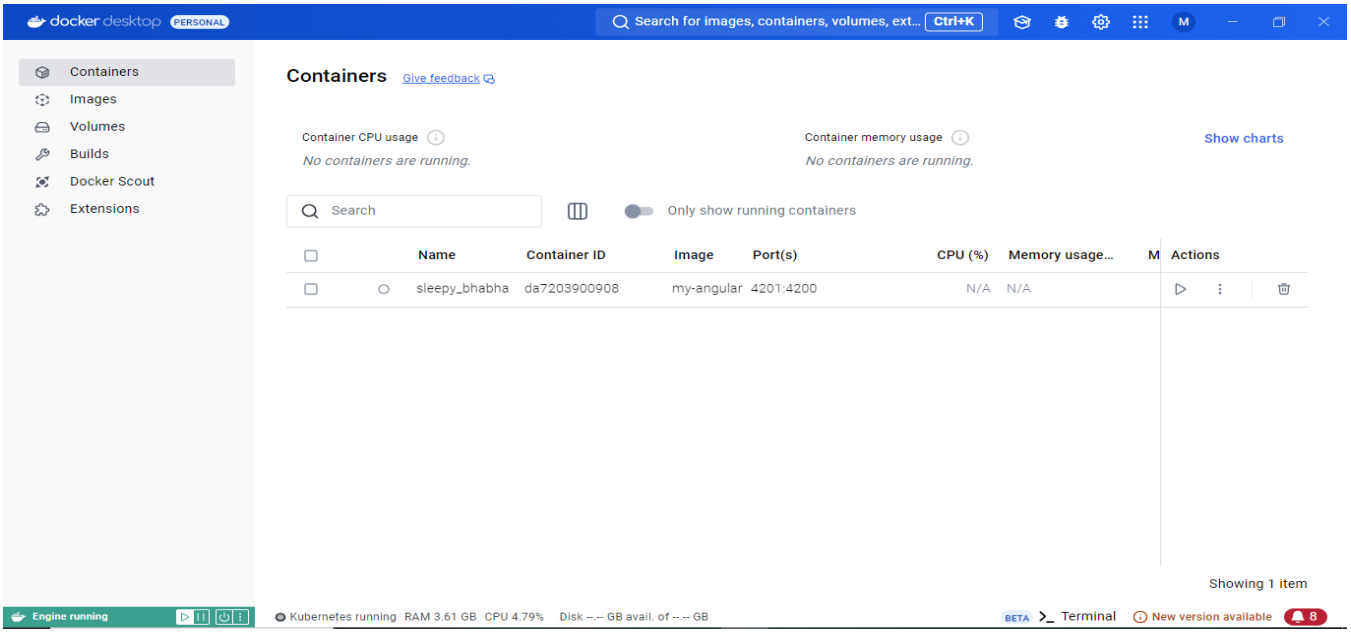


- **Etape 2 :**

Ensuite, nous allons taper la commande "docker run –d –it –p 4201:4200 my-angulardev" qui nous permet d'exécuter un conteneur à partir de l'image Docker nommée «myangular-dev». Nous avons donc notre conteneur appelé «sleepy_bhabha» présent sur notre Docker Desktop.

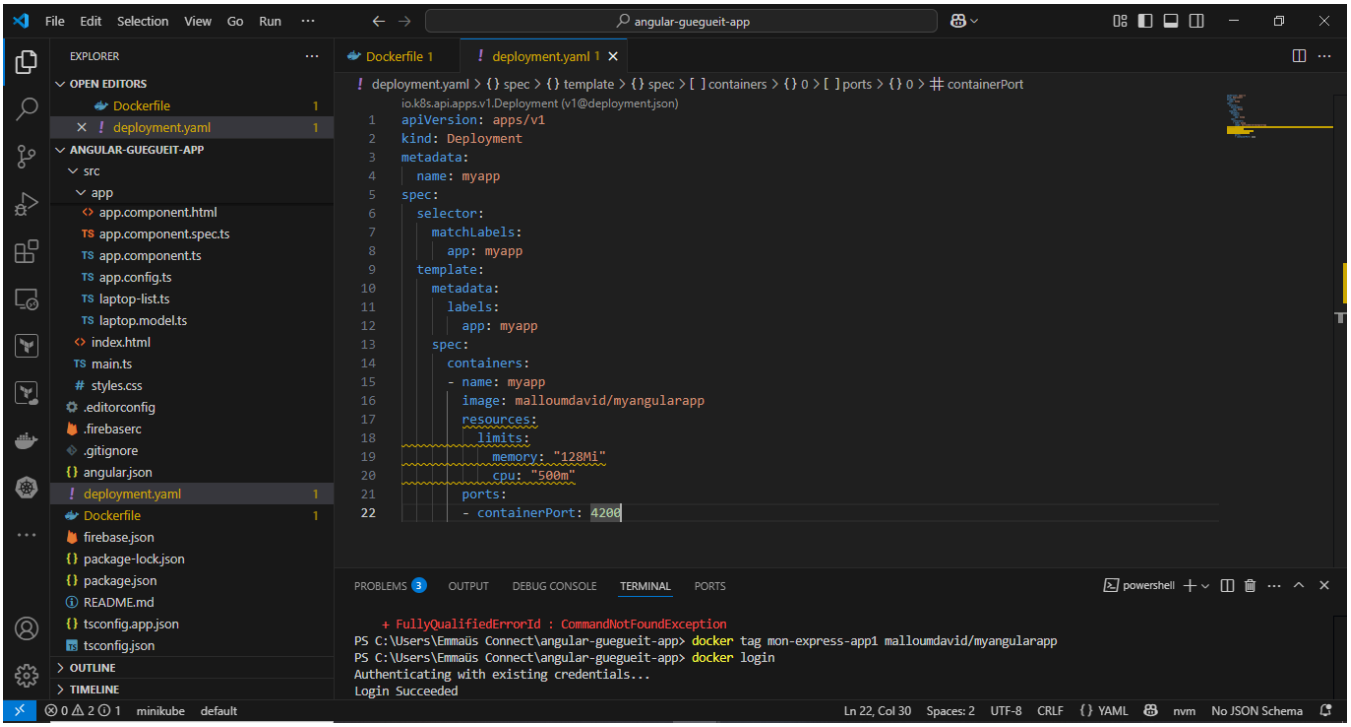


Nous avons donc notre conteneur appelé «sleepy_bhabha» présent sur notre Docker Desktop.



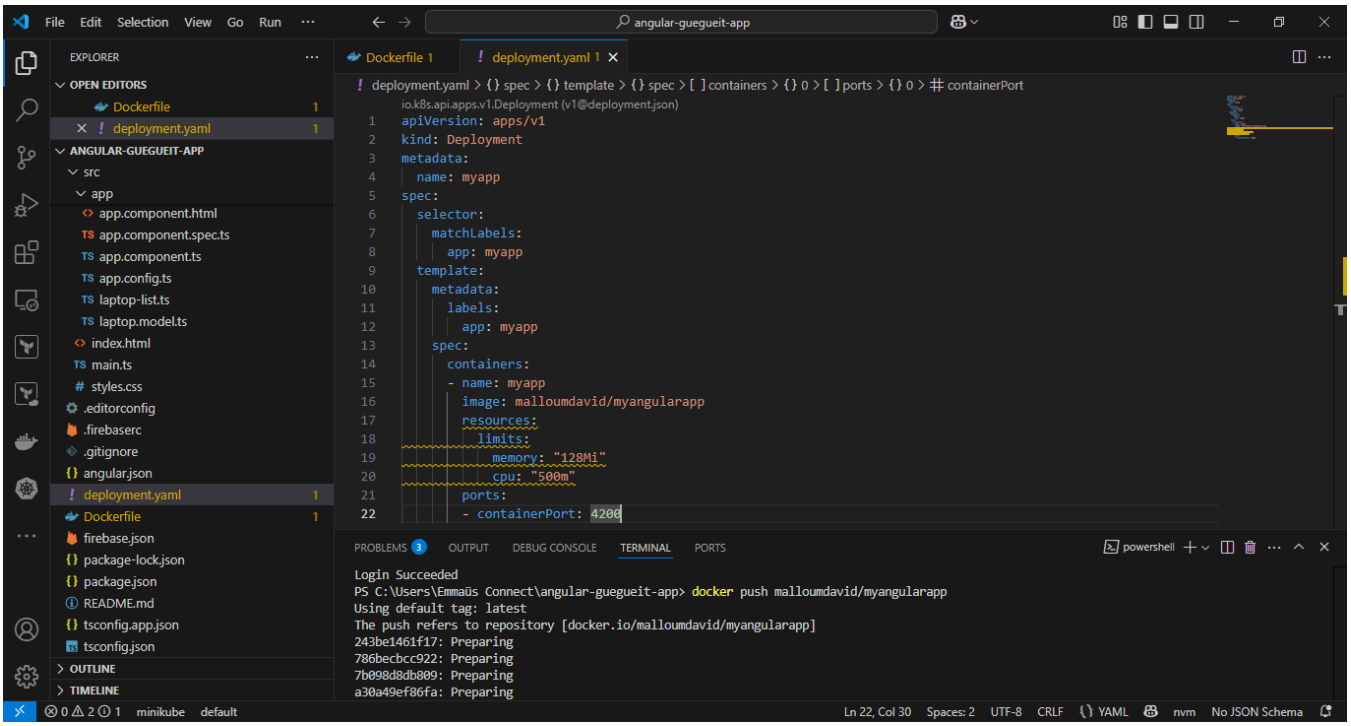
- **Etape 3 :**

Ensuite, nous allons créer le fichier deployment.yaml et taper les commandes "docker tag mon-express-app1 malloumdavid/myangularapp" qui nous permet d'ajouter un nouveau nom (tag) à une image Docker existante et "docker login" qui nous permet de nous authentifier auprès de notre Docker Hub.

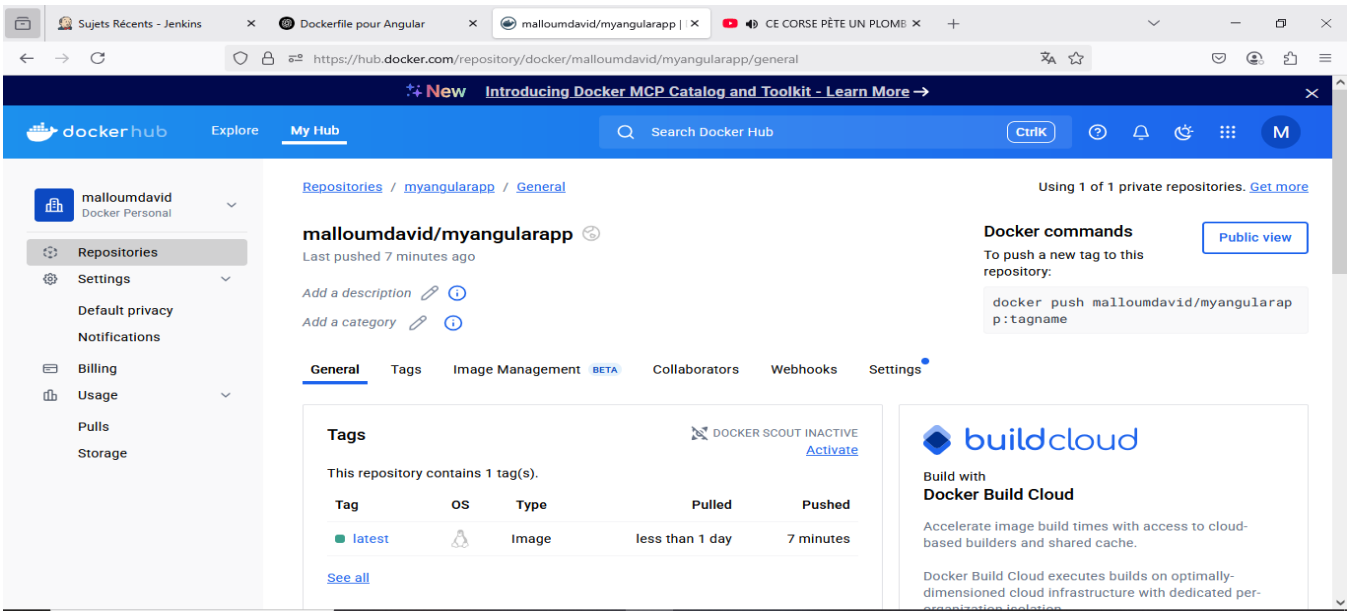


- **Etape 4 :**

Ensuite, nous allons taper la commande "docker push malloumdavid/myangularapp" qui nous permet d'effectuer le push de l'image Docker locale vers notre Docker Hub. Nous avons donc notre image docker présent sur notre Docker Hub.

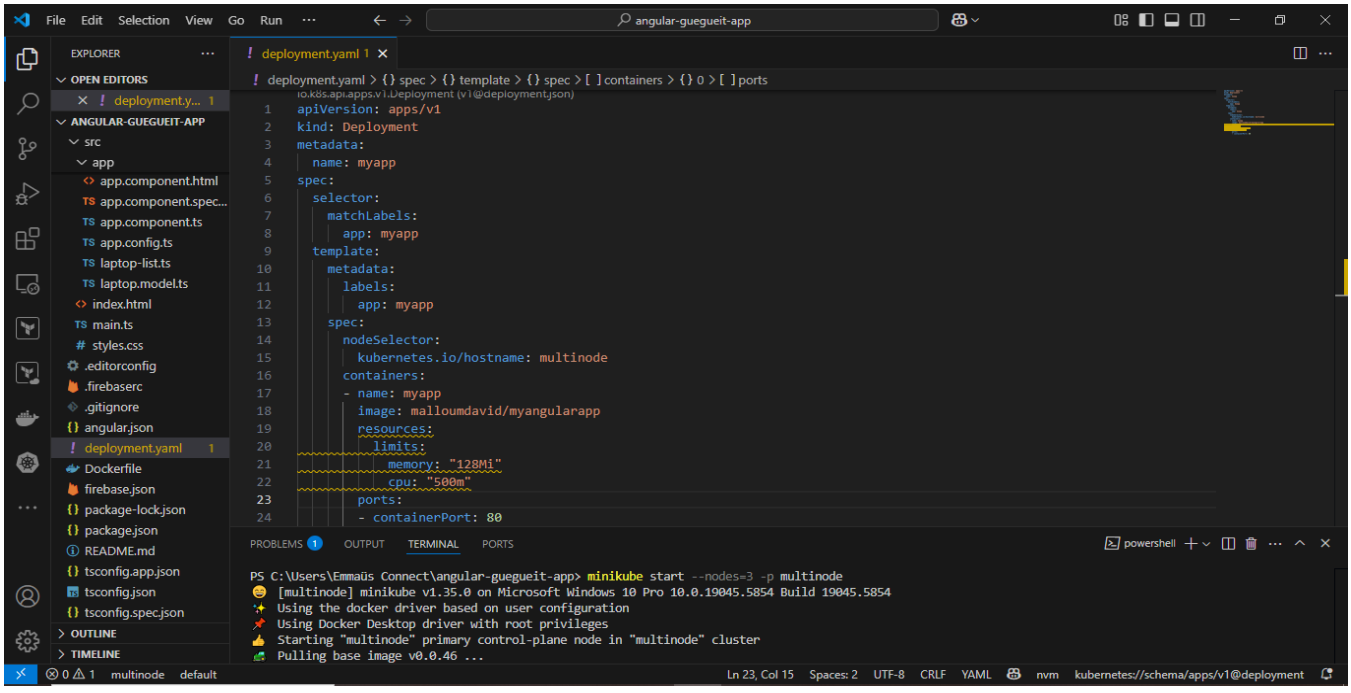


Nous avons donc notre image docker présent sur notre Docker Hub.



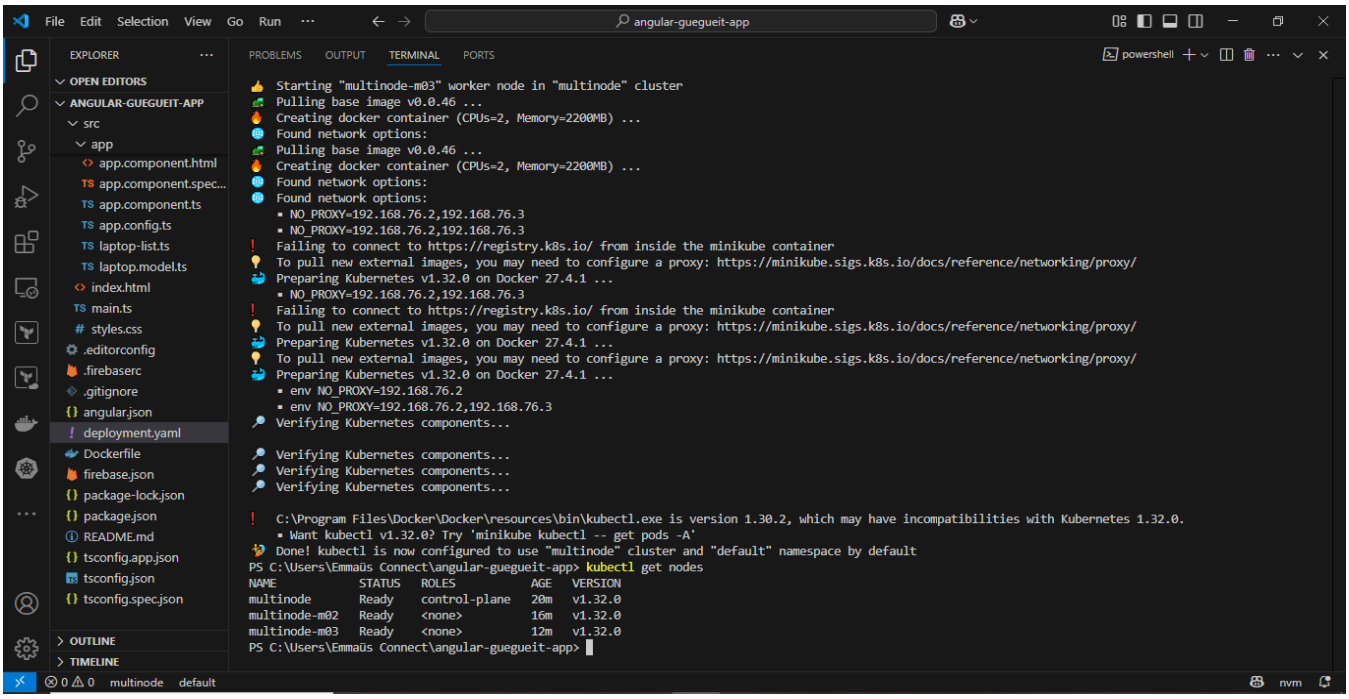
- **Etape 5 :**

Ensuite, nous allons modifier le fichier deployment.yaml et taper la commande "minikube start - -nodes=3 –p multinode" qui nous permet de démarrer un cluster Kubernetes local avec Minikube, en spécifiant les 3 nœuds dans le cluster (1 master + 2 workers) et un profil nommé «multinode».

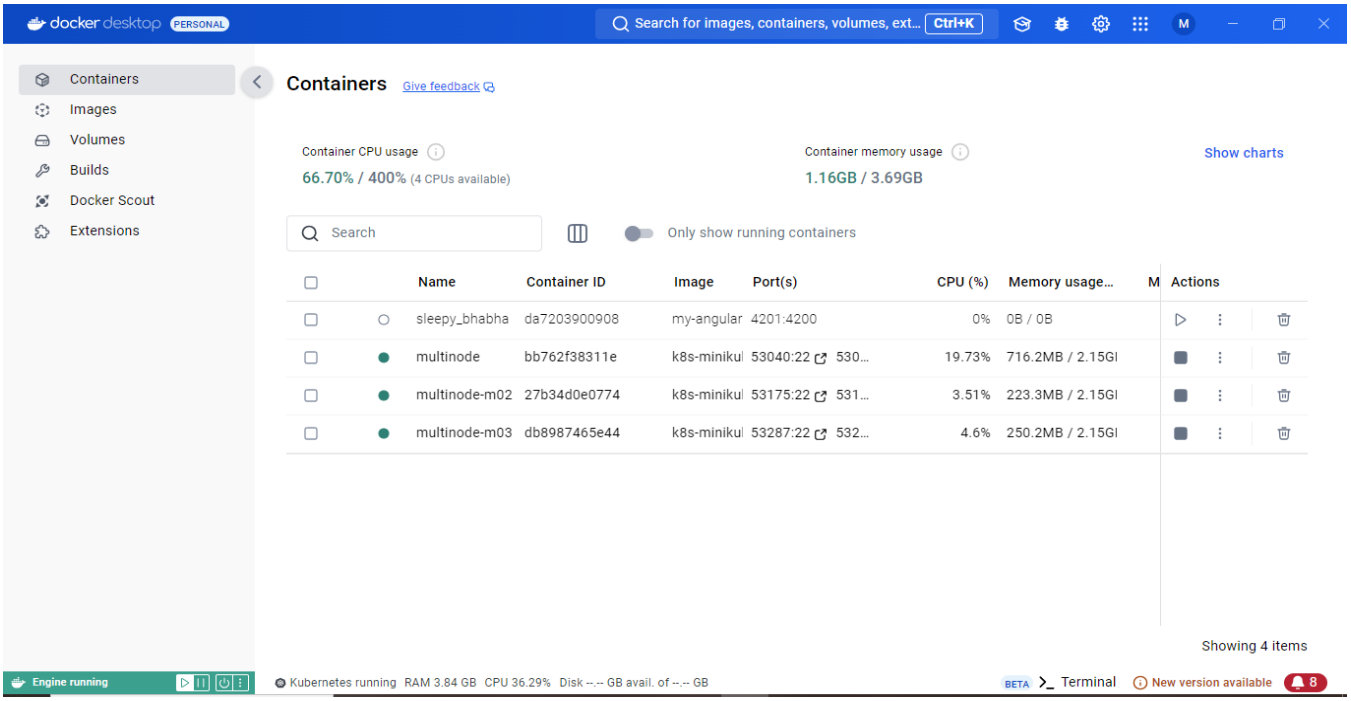


- **Etape 6 :**

Et enfin, nous allons taper la commande "kubectl get nodes" qui nous permet lister tous les nœuds d'un cluster Kubernetes, avec des informations de base sur chacun d'eux. Nous avons donc les 3 nœuds présents sur notre Docker Desktop.



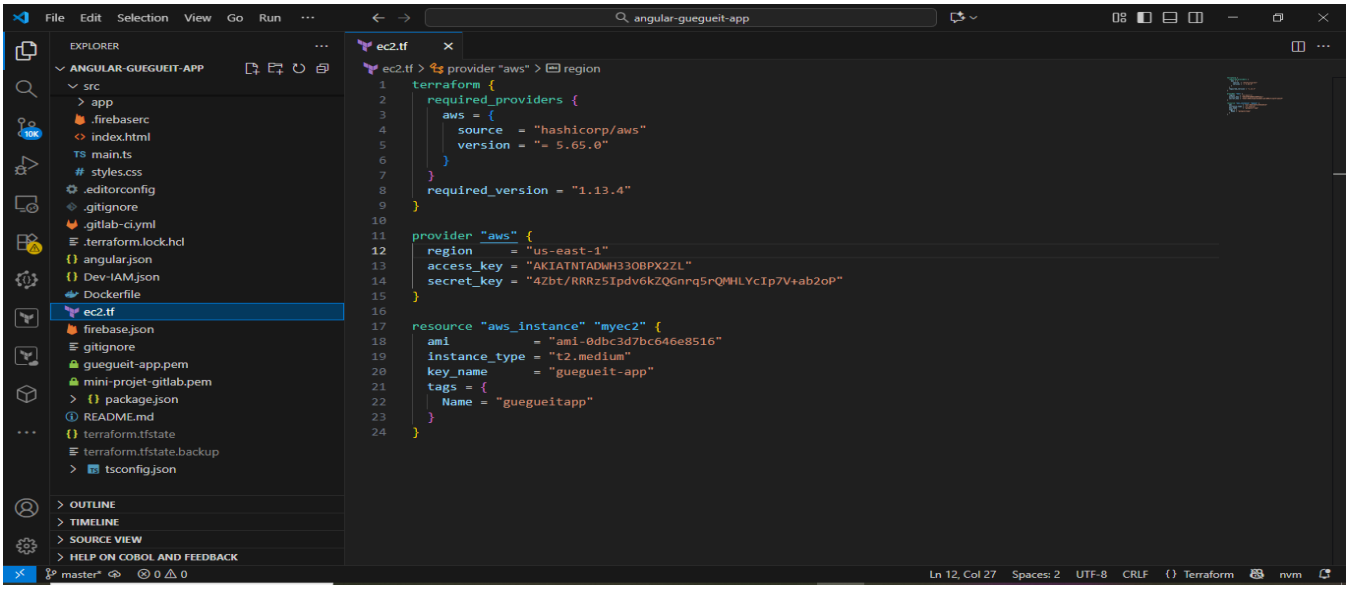
Nous avons donc les 3 nœuds présents sur notre Docker Desktop.



Création des instances ec2 avec Terraform

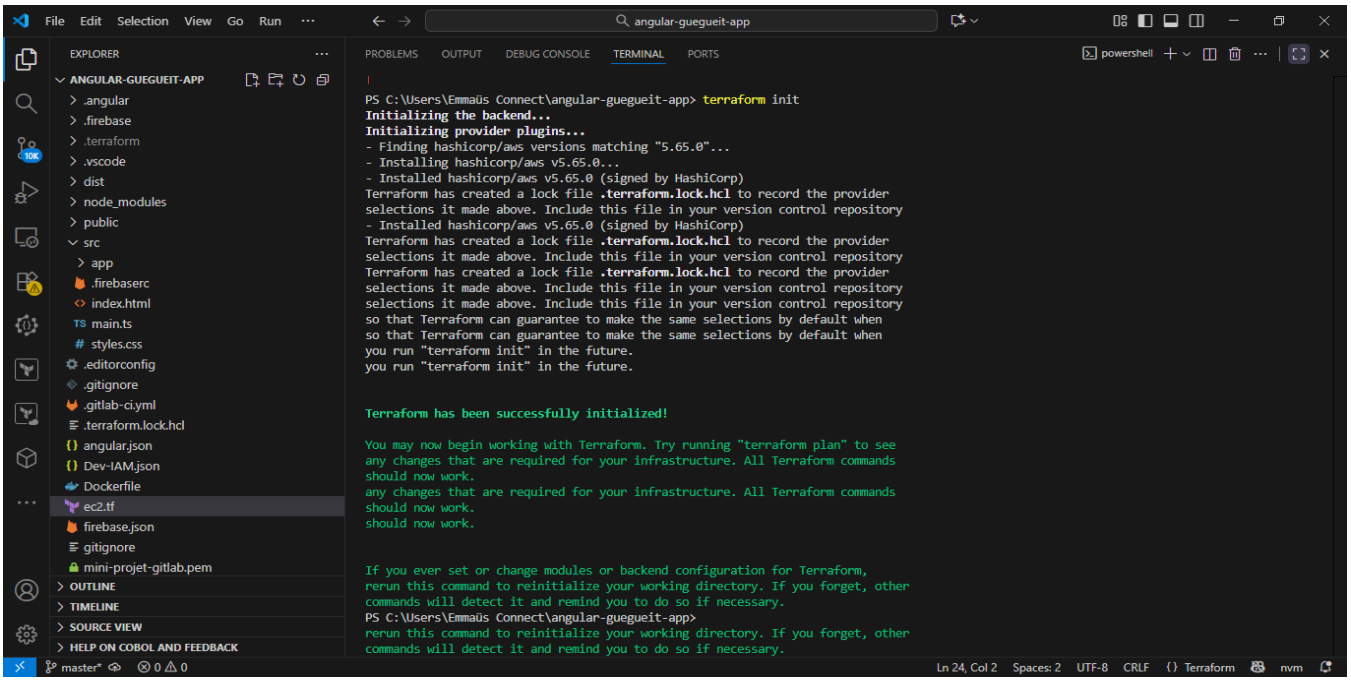
- **Etape 1 :**

Nous allons créer et configurer le fichier ec2.tf



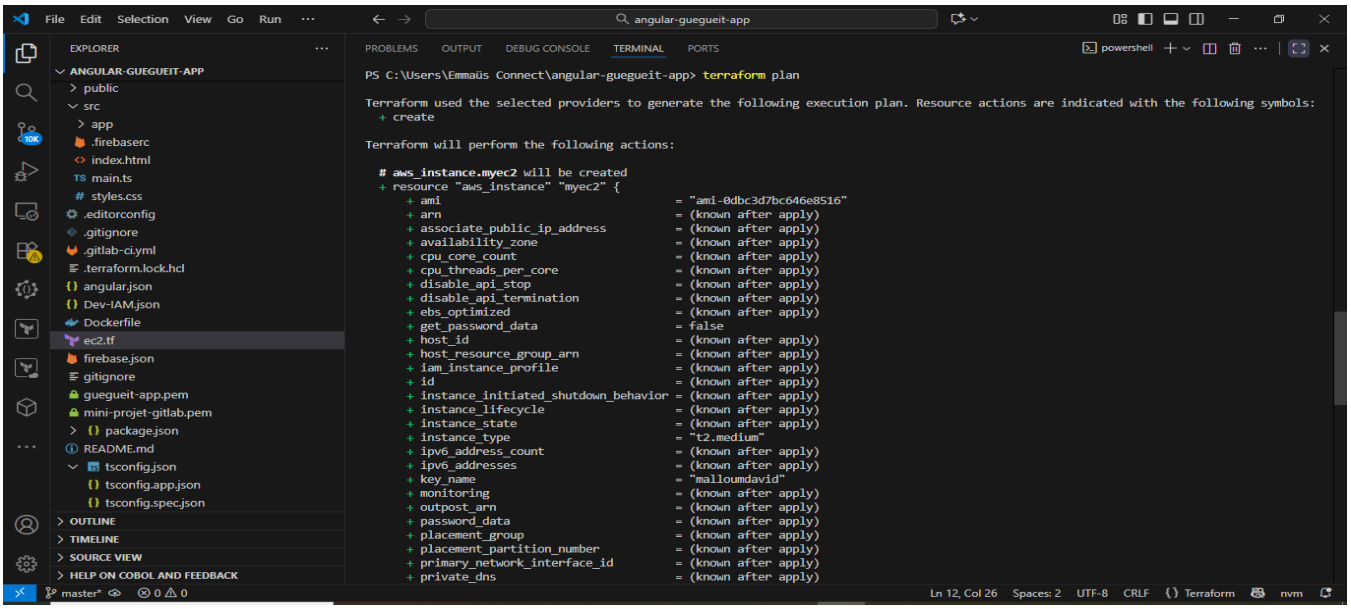
- **Etape 2 :**

Ensuite, au niveau du terminal, nous allons taper la commande « terraform init » qui nous permet d'initialiser le répertoire de travail contenant les fichiers de configuration Terraform (.tf).



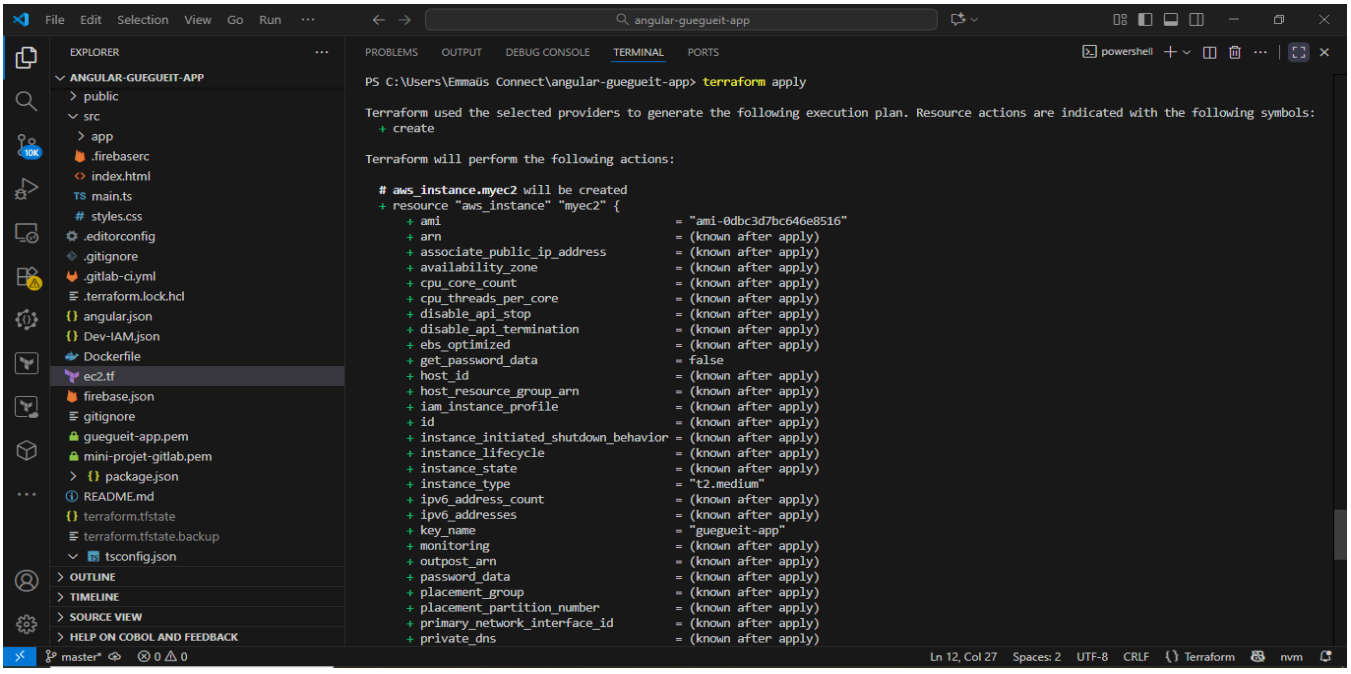
- **Etape 3 :**

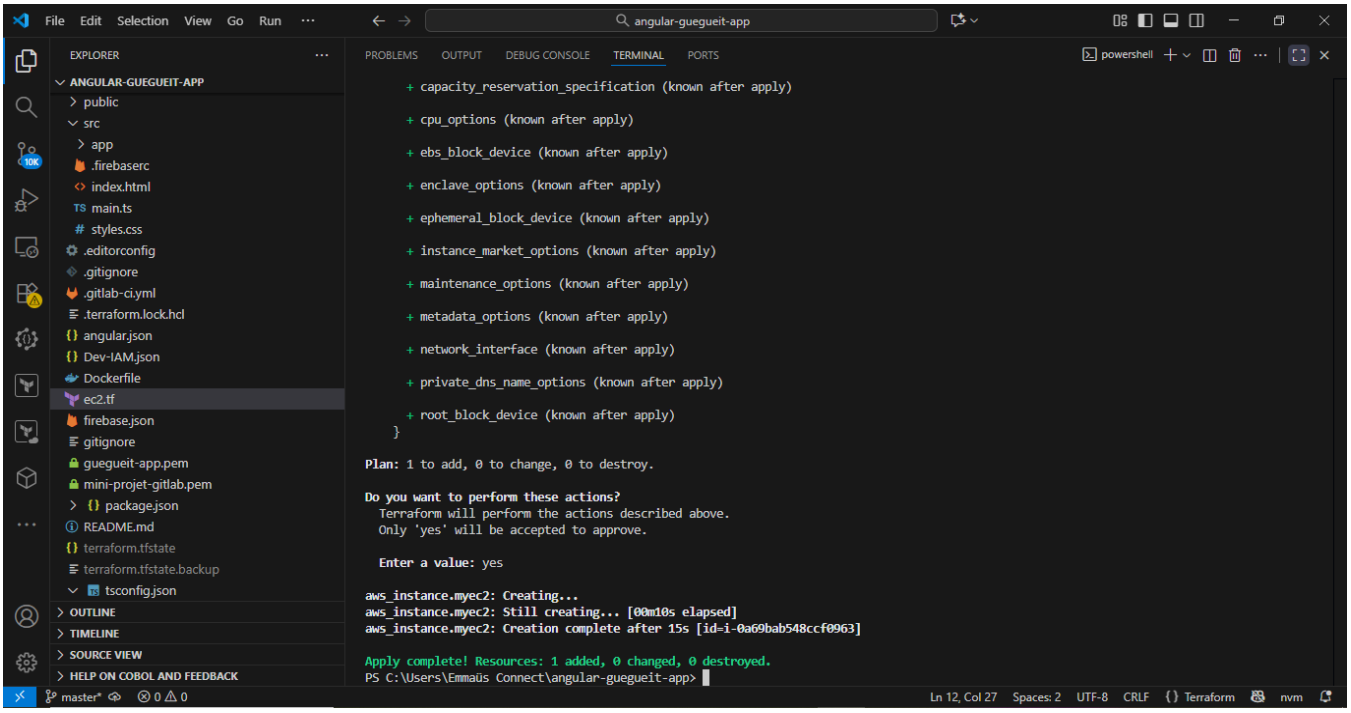
Ensuite, nous allons taper la commande « terraform plan » qui nous permet de prévisualiser les changements que Terraform va appliquer à notre infrastructure, sans les exécuter réellement.



- **Etape 3 :**

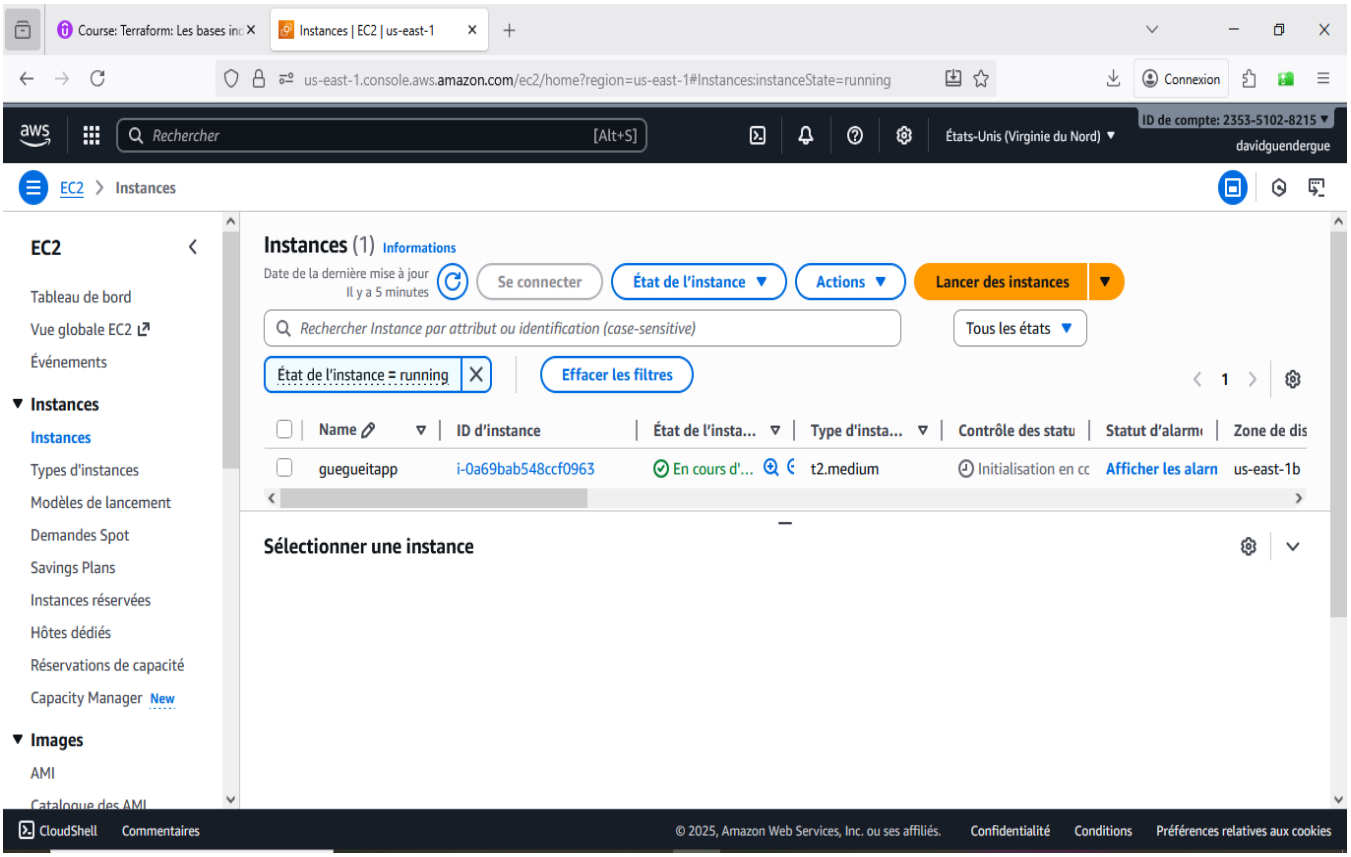
Ensuite, nous allons taper la commande « terraform apply » qui nous permet d'exécuter réellement les changements sur notre infrastructure.





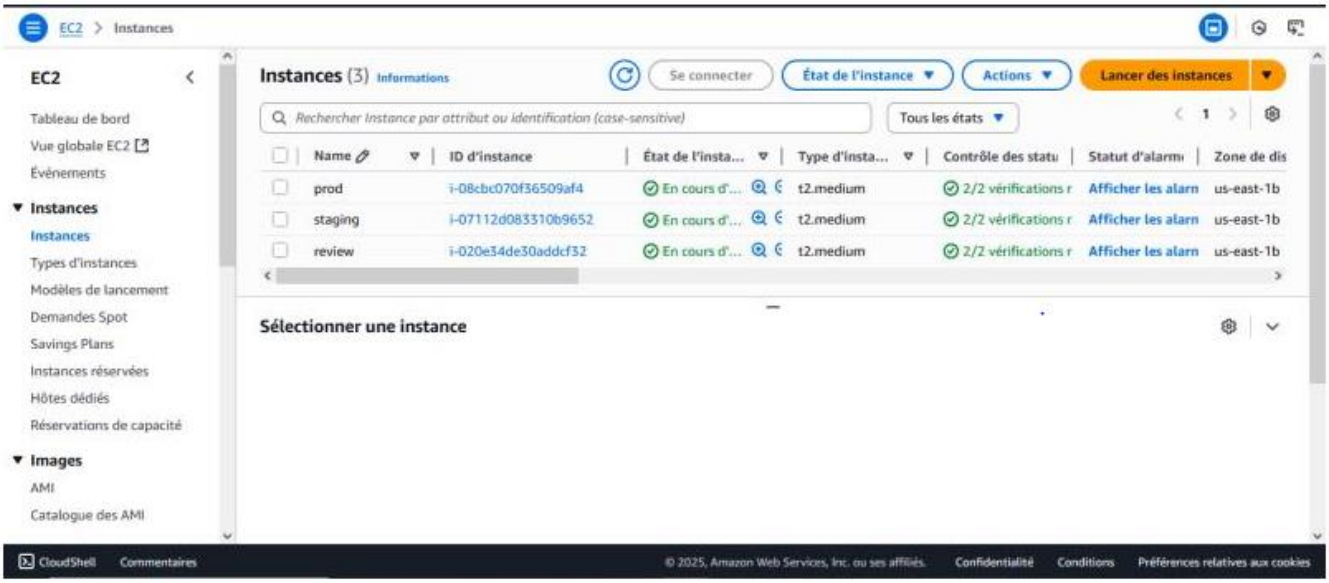
- **Etape 4 :**

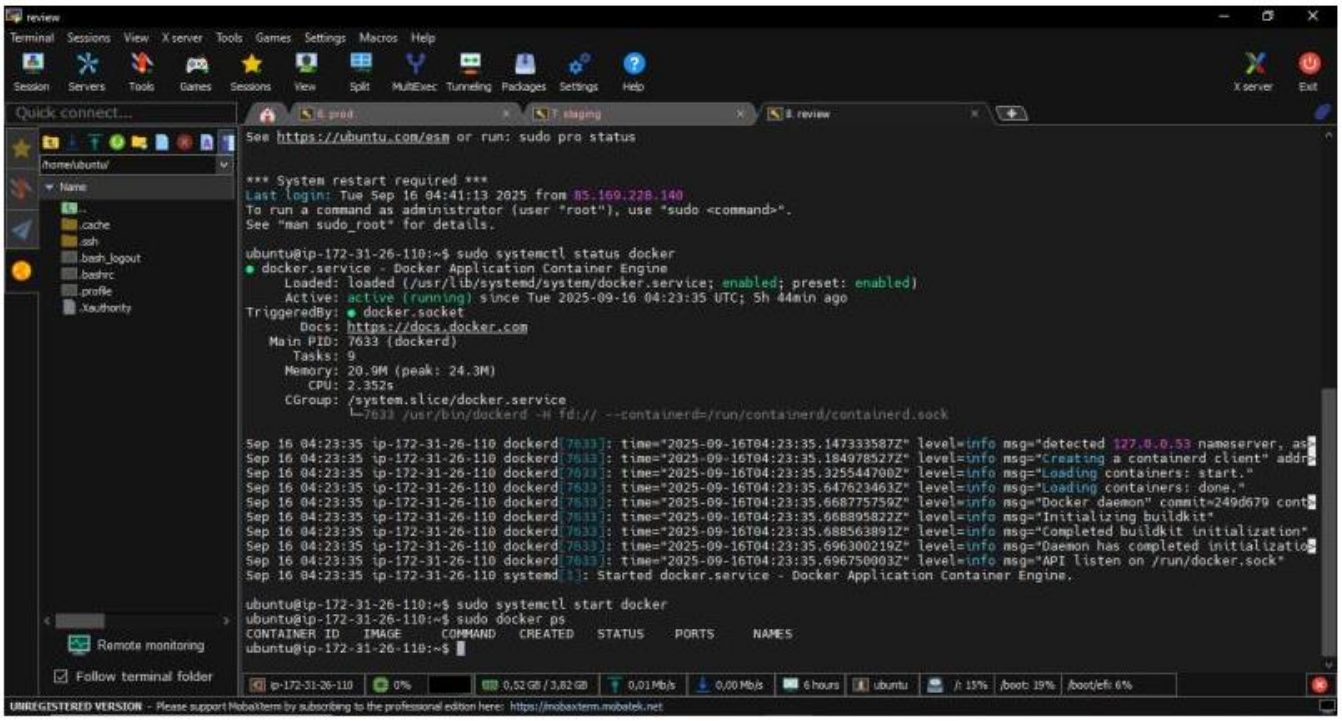
Et enfin, nous allons pouvoir apercevoir notre instance ec2 créée sur la console de notre compte AWS.

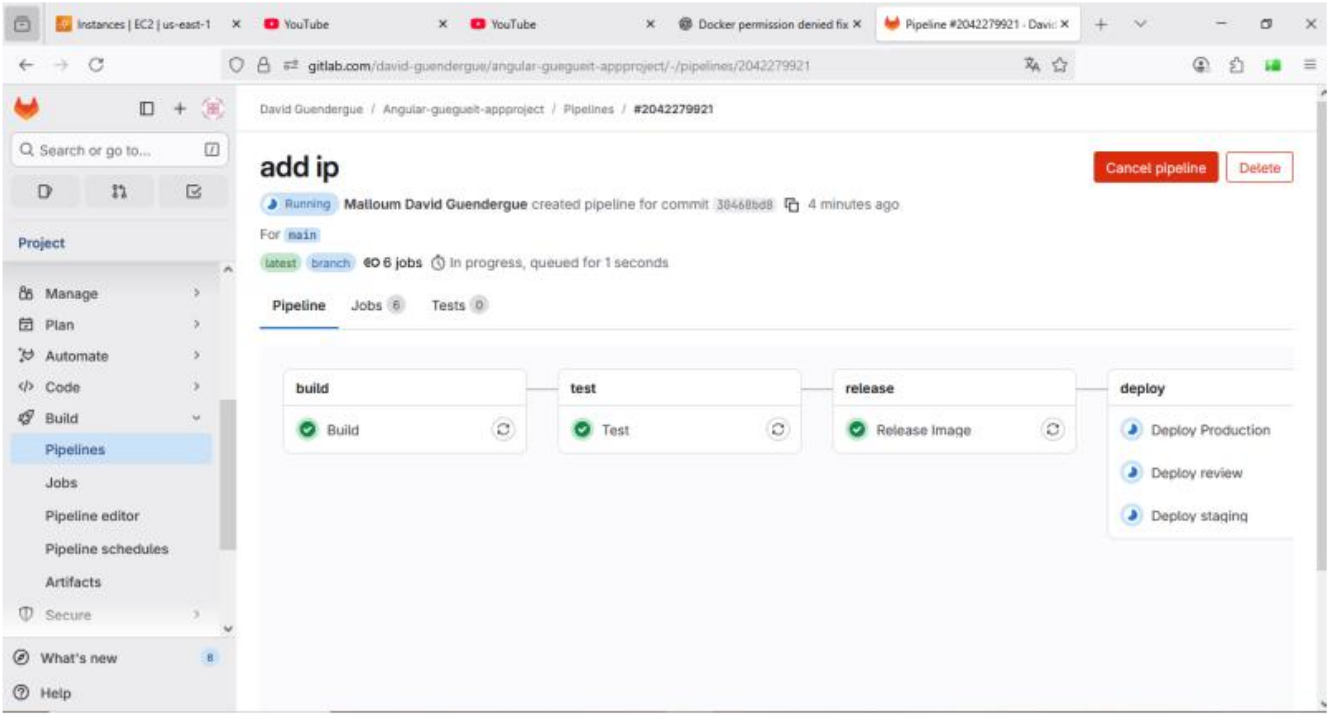


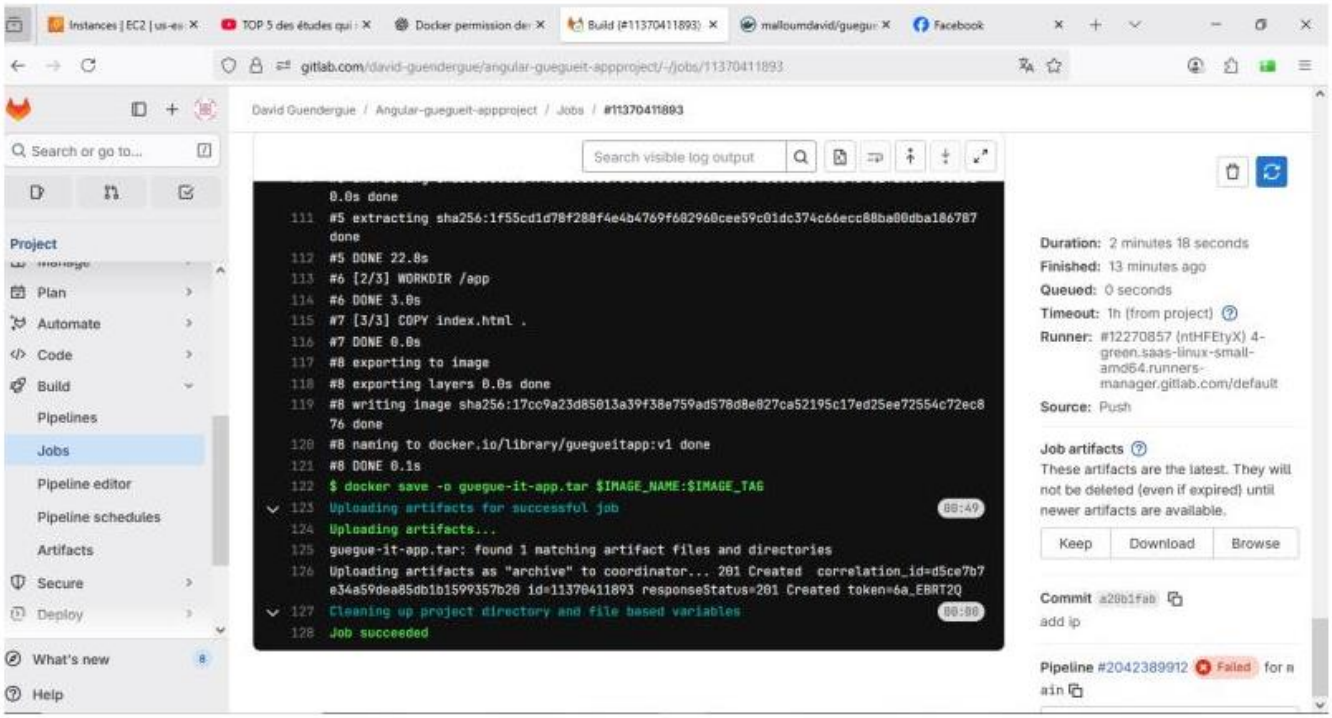
Mise en place d’un pipeline CI/CD avec GitLab

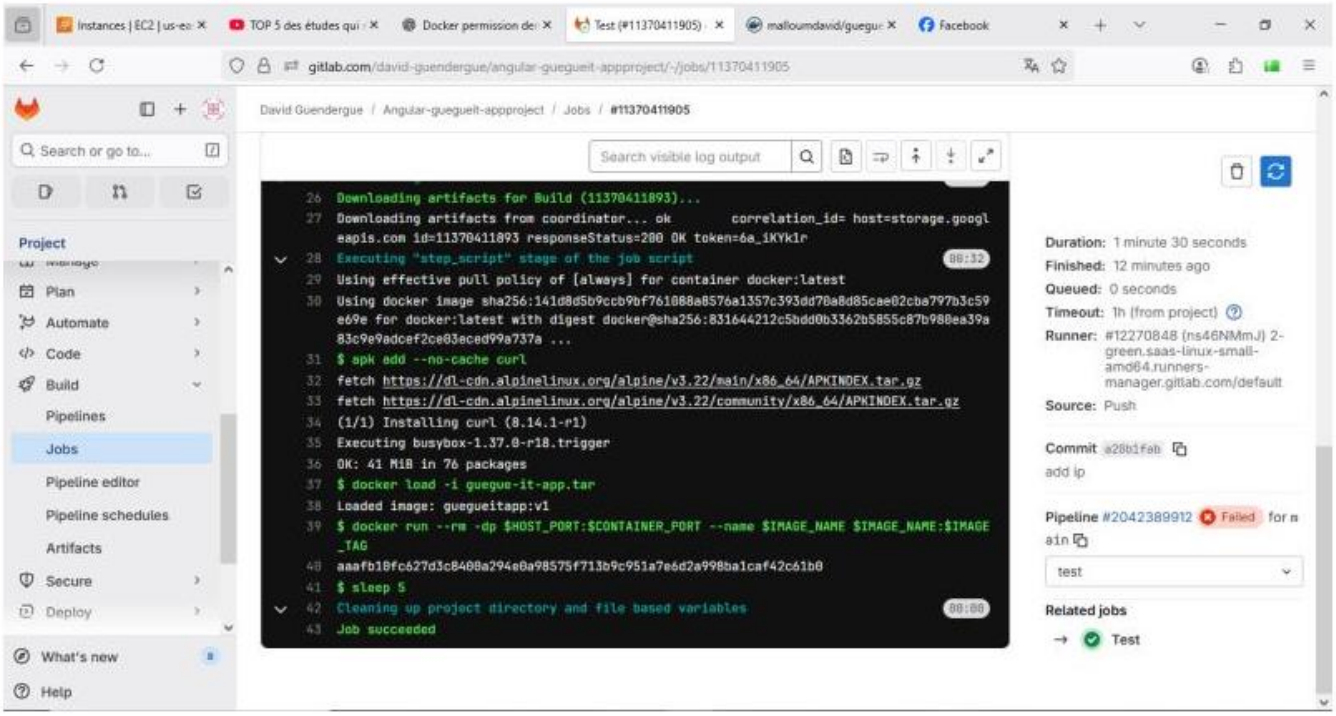
Nous avons tout d'abord créé 3 instances ec2 (prod, staging et review) sur AWS, ensuite nous avons visualiser ces instances ec2 sur MobaXterm, ensuite nous avons utilisé GitLab pour transformer les configurations du DockerFile obtenu en langage shell, effectuer le push de notre code sur notre GitHub et sur notre GitLab, effectuer des configurations de sécurité avec notre Docker Hub et sur base64 pour la clé publique et enfin nous avons modifié l'éditeur de pipeline en fonction des résultats obtenus afin de cliquer sur le bouton «Commit changes».

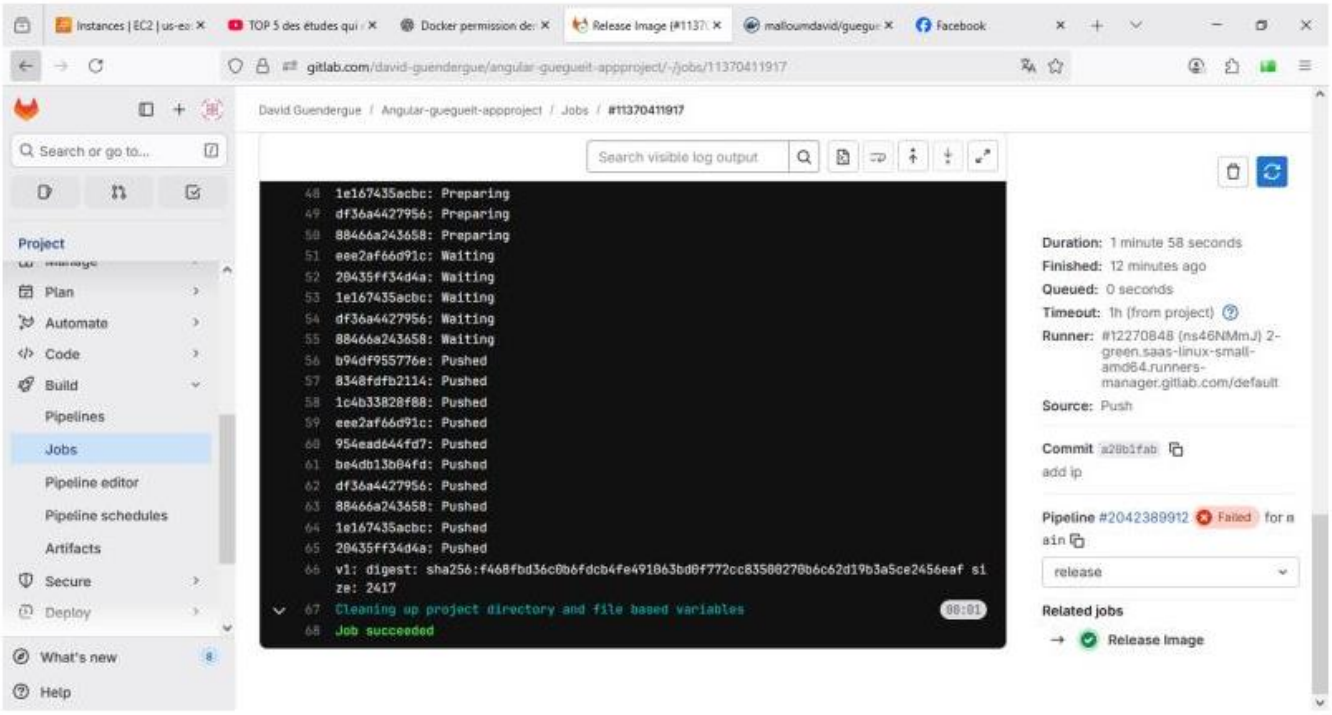


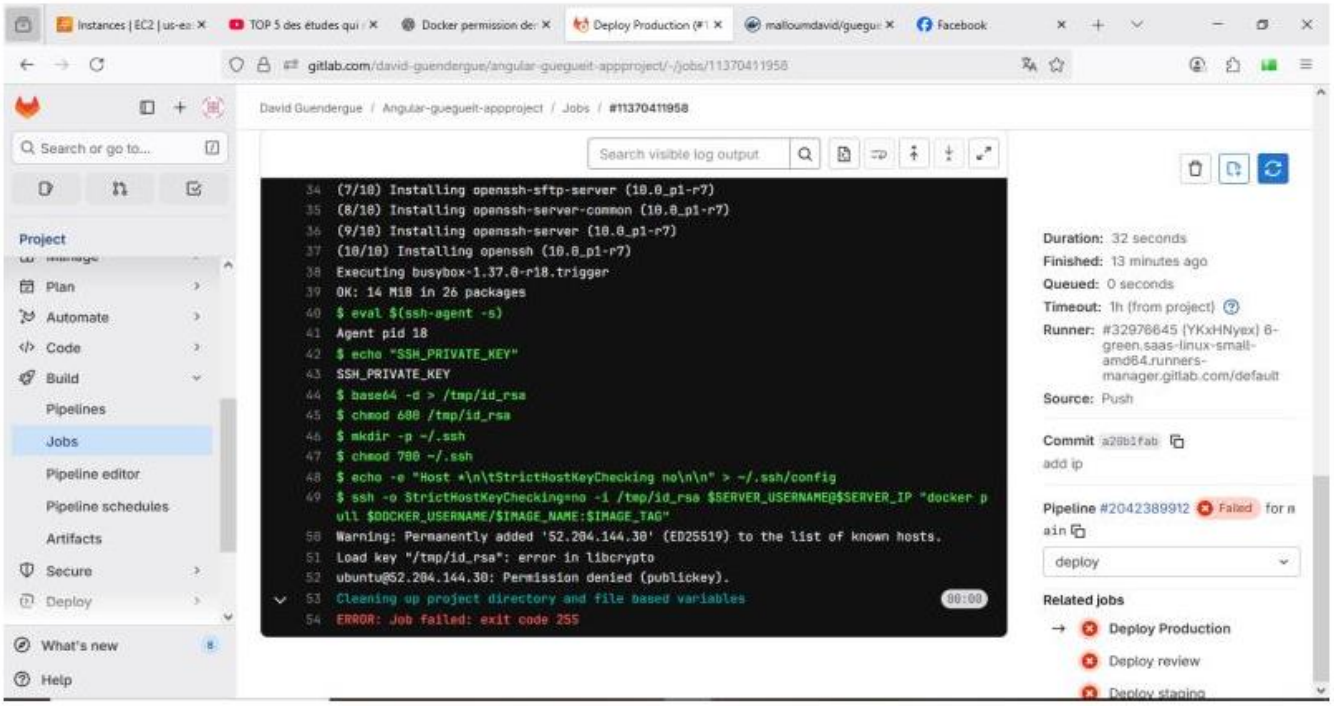



gitlab.com/david-guendergue/angular-guegueit-appproject/-/pipelines/2042279921
David Guendergue / Angular-guegueit-appproject / Pipelines / #2042279921
add ip
Cancel pipeline
Delete
Running Malloum David Guendergue created pipeline for commit 38468bd8 4 minutes ago
For main
latest branch 6 jobs In progress, queued for 1 seconds
Pipeline Jobs 6 Tests 0
build
Build
test
Test
release
Release Image
deploy
Deploy Production
Deploy review
Deploy staging
Project
Manage
Plan
Automate
Code
Build
Pipelines
Jobs
Pipeline editor
Pipeline schedules
Artifacts
Secure
What's new
Help


gitlab.com/david-guendergue/angular-guegueit-appproject/-/jobs/11370411893
David Guendergue / Angular-guegueit-appproject / Jobs / #11370411893
Search visible log output
0.0s done
#5 extracting sha256:1f55cd1d78f288f4e4b4769f602960cee59c01dc374c66ecc88ba00dba186787 done
#5 DONE 22.8s
#6 [2/3] WORKDIR /app
#6 DONE 3.0s
#7 [3/3] COPY index.html .
#7 DONE 0.0s
#8 exporting to image
#8 exporting layers 0.0s done
#8 writing image sha256:17cc9a23d85013a39f38e759ad578d8e827ca52195c17ed25ee72554c72ec876 done
#8 naming to docker.io/library/guegueitapp:v1 done
#8 DONE 0.1s
$ docker save -o guegue-it-app.tar $IMAGE_NAME:$IMAGE_TAG
Uploading artifacts for successful job 00:49
Uploading artifacts...
guegue-it-app.tar: found 1 matching artifact files and directories
Uploading artifacts as "archive" to coordinator... 201 Created correlation_id=d5ce7b7e34a59dea85db1b1599357b20 id=11370411893 responseStatus=201 Created token=6a_EBRT2Q
Cleaning up project directory and file based variables 00:00
Job succeeded
Duration: 2 minutes 18 seconds
Finished: 13 minutes ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #12270857 (ntHFEtyX) 4-green.saas-linux-small-amd64.runners-manager.gitlab.com/default
Source: Push
Job artifacts
These artifacts are the latest. They will not be deleted (even if expired) until newer artifacts are available.
Keep
Download
Browse
Commit a20b1fab
add ip
Pipeline #2042389912 Failed for main
Project
Plan
Automate
Code
Build
Pipelines
Jobs
Pipeline editor
Pipeline schedules
Artifacts
Secure
Deploy
What's new
Help

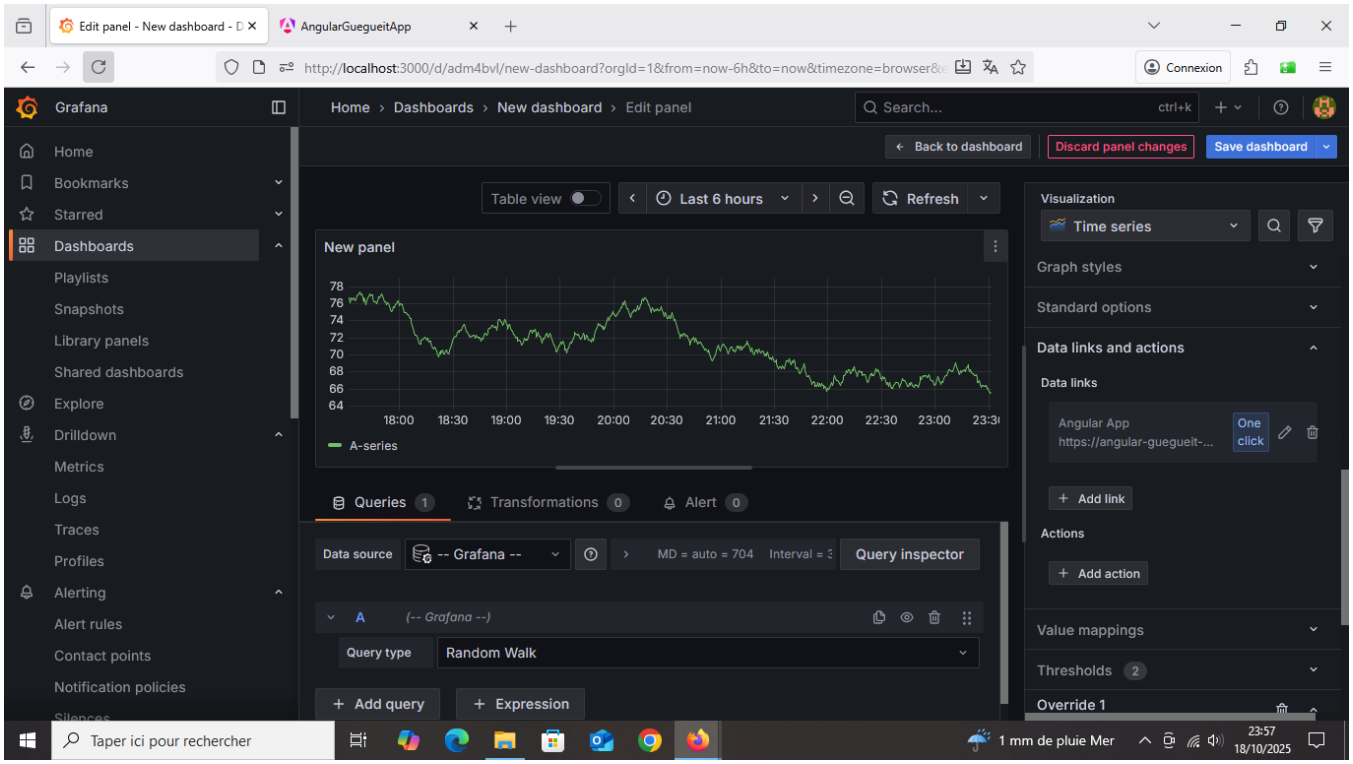

gitlab.com/david-guendergue/angular-guegueit-appproject/-/jobs/11370411905
David Guendergue / Angular-guegueit-appproject / Jobs / #11370411905
Search visible log output
Downloading artifacts for Build (11370411893)...
Downloading artifacts from coordinator... ok correlation_id= host=storage.googleapis.com id=11370411893 responseStatus=200 OK token=6a_iKYk1r
Executing "step_script" stage of the job script
Using effective pull policy of [always] for container docker:latest
Using docker image sha256:141d8d5b9ccb9bf761088a8576a1357c393dd70a8d85cae02cba797b3c59e69e for docker:latest with digest docker@sha256:831644212c5bdd0b3362b5855c87b988ea39a83c9e9adcef2ce03eced99a737a ...
$ apk add --no-cache curl
fetch https://dl-cdn.alpinelinux.org/alpine/v3.22/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.22/community/x86_64/APKINDEX.tar.gz
(1/1) Installing curl (8.14.1-r1)
Executing busybox-1.37.0-r18.trigger
OK: 41 MiB in 76 packages
$ docker load -i gueguе-it-app.tar
Loaded image: guegueitapp:v1
$ docker run --rm -dp $HOST_PORT:$CONTAINER_PORT --name $IMAGE_NAME $IMAGE_NAME:$IMAGE_TAG
aaafb10fc627d3c8408a294e0a98575f713b9c951a7e6d2a998ba1caf42c61b0
$ sleep 5
Cleaning up project directory and file based variables
Job succeeded
Duration: 1 minute 30 seconds
Finished: 12 minutes ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #12270848 (ns46NMmJ) 2-green.saas-linux-small-amd64.runners-manager.gitlab.com/default
Source: Push
Commit a20b1fab
add ip
Pipeline #2042389912 Failed for main
test
Related jobs
Test


gitlab.com/david-guendergue/angular-guegueit-appproject/-/jobs/11370411917
David Guendergue / Angular-guegueit-appproject / Jobs / #11370411917
Search visible log output
1e167435acbc: Preparing
df36a4427956: Preparing
88466a243658: Preparing
eee2af66d91c: Waiting
20435ff34d4a: Waiting
1e167435acbc: Waiting
df36a4427956: Waiting
88466a243658: Waiting
b94df955776e: Pushed
8348fdfb2114: Pushed
1c4b33820f88: Pushed
eee2af66d91c: Pushed
954ead644fd7: Pushed
be4db13b04fd: Pushed
df36a4427956: Pushed
88466a243658: Pushed
1e167435acbc: Pushed
20435ff34d4a: Pushed
v1: digest: sha256:f468fbd36c0b6fdcb4fe4910b3bd0f772cc83500270b6c62d19b3a5ce2456eaf size: 2417
Cleaning up project directory and file based variables
Job succeeded
Duration: 1 minute 58 seconds
Finished: 12 minutes ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #12270848 (ns46NMmJ) 2-green.saas-linux-small-amd64.runners-manager.gitlab.com/default
Source: Push
Commit a20b1fab
add ip
Pipeline #2042389912 Failed for main
release
Related jobs
Release Image



E- <u>Observabilité & opérations (dashboards, alertes, runbooks)</u> :

Nous avons réalisé nos travaux d'observabilité et d'opérations de notre application sur Grafana comme le montre la capture d'écran ci-dessous de notre graphe.



F- Tests & résultats (méthode, outils, mesures, limites) :

Pour les tests et les résultats, nous avons :

- La méthode utilisée est la culture DevOps.
- Les outils utilisés sont Angular, AWS, Docker, Minikube, GitLab, Terraform, Firebase, Grafana et AWS Pricing Calculator.
- Les mesures utilisées pour la création des instances ec2 sont par exemple le type d'instance « t2.medium » et la région « us-east-1 ».
- La limite principale que nous avons rencontrée est au niveau des 3 stages de Deploy sur le pipeline CI/CD de GitLab qui n'a pas pu se réaliser avec succès.

G- Coûts & optimisation (FinOps) :

- Nous utilisons 3 instances ec2 t2.medium en mode On-Demand (24/7) et nous utilisons « AWS Pricing Calculator » pour effectuer nos calculs. La région choisie est : US East (N. Virginia) (us-east-1)

- En calculant le coût mensuel total en On-Demand, nous avons :

  Coût mensuel total = 33,87 USD/mois

- En estimant le coût mensuel avec des Reserved Instances, nous avons :

  Coût mensuel avec des Reserved Instances = 20,95 USD/mois

- En estimant le coût mensuel avec un Compute Saving Plan, nous savons :

  Coût mensuel avec un Compute Saving Plan = 24,16 USD/mois

- En comparant les 3 options dans un tableau (Type, Coût mensuel, Économie vs OnDemand), nous pouvons conclure que la meilleure stratégie FinOps est la stratégie des Reserved Instances parce qu'elle est la stratégie ayant le coût le moins cher et le plus économique parmi les 3 options disponibles.

CONCLUSION (Risques, limites, perspectives)

En définitive, nous pouvons affirmer que le choix du Framework Angular pour développer l'application était la meilleure option parce qu'il est basé sur le langage JavaScript qu'on peut utiliser en Front end comme en Back end, et il nous permet également de consommer moins de ressources en production par rapport à une même application écrit en langage PHP par exemple. Un logiciel fiable, ce n'est pas juste du code qui tourne sans erreur. C'est un produit qui protège les données, contrôle qui peut y accéder et détecte les incidents avant qu'ils ne dégénèrent. Les infrastructures informatiques mal gérées génèrent des problématiques sérieuses qui impactent directement la performance et la sécurité d'une entreprise. En effet, des pannes système, des performances limitées, des pertes de données, ainsi que des lenteurs de connexion peuvent survenir. L'hétérogénéité des technologies, la diversité des équipements (serveurs, réseaux, systèmes de stockage), ainsi que le manque de compétences techniques peuvent créer une surcharge de travail difficile à gérer. Au fil du temps, les coûts liés à la maintenance et à la gestion des infrastructures deviennent de plus en plus élevés. Dans ce contexte, confier cette gestion à un prestataire spécialisé en gestion d'infrastructures IT s'avère être une solution stratégique et plus rentable.

REFERENCES & ANNEXES

- Jacques Printz (préf. Yves Caseau), Architecture logicielle : concevoir des applications simples, sûres et adaptables, Paris, Dunod, 2012, 3e éd., 512 p.
- Joëlle Delacroix et Alain Cazes, *Architecture des machines et des systèmes informatiques*, Paris, Dunod, 2018, 6e éd., 560 p.
- https://angular.dev/
- https://aws.amazon.com/fr/
- https://www.docker.com/
- https://minikube.sigs.k8s.io/docs/start/?arch=%2Fwindows%2Fx86-64%2Fstable%2F.exe+download
- https://about.gitlab.com/
- https://www.hashicorp.com/fr/products/terraform?utm_source=google&utm_channel_bucket=paid&utm_medium=sem&utm_campaign=Speed_EMEA_ENG_BOFU_Practitioner_SEM_Brand-Terraform_ILM&utm_content=terraform%20cloud-144080651246-645564370290&utm_offer=signup&gad_source=1&gad_campaignid=19592887117&gclid=CjwKCAjwjffHBhBuEiwAKMb8pEi-pSrJ-Lulw-G1uptkOQngLT_87yOrzwwkeO462orfwPIl-BxTIBoCO9MQAvD_BwE
- https://firebase.google.com/
- https://grafana.com/
- https://calculator.aws/#/