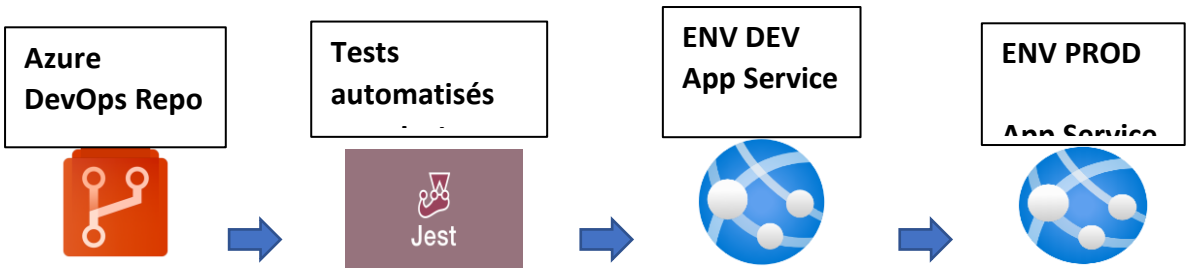
Admin Infra : Labo 10

Exercice 1 : Pipeline Exoplanets

Nous allons créer un premier pipeline permettant de déployer le site Web des exoplanètes. Ce site Web utilise une base de données PostgreSQL. Pour le déploiment de ce pipeline, nous utiliserons Azure DevOps.

Le pipeline est le suivant :



Conventions de nommage :

xxx est à remplacer par un terme unique. Par exemple la première lettre de votre prénom suivi des 2 lettres de votre nom.

1. **exoapp-xxx-dev** désigne l'environnement de développement créé sur le portal Azure (https://portal.azure.com)
2. **exoapp-xxx-prod** désigne l'environnement de production créé sur le portal Azure
3. **exoapp-xxx-db-dev** désigne la base de données de l'environnement de développement créé sur ElephantSQL
4. **exoapp-xxx-db-prod** désigne la base de données de l'environnement de production créé sur ElephantSQL

Objectifs du pipeline :

1. Le dépôt Azure DevOps est composé des sources du site des exoplanètes disponible sur MooVin.
2. L'environnement de développement sera une App Service node.js version 16 LTS (**exoapp-xxx-dev**) hébergée sur https://portal.azure.com et contenant une base de données ElephantSQL
3. L'environnement de développement sera une App Service node.js version 16 LTS (**exoapp-xxx-prod**) hébergée sur https://portal.azure.com et contenant une base de données ElephantSQL
4. **Les bases de données dev et prod sont distinctes**

5. A chaque commit/push sur le dépôt, les tests seront effectués **et si les tests passent le code sera déployé automatiquement dans l'environnement dev**
6. L'environnement prod est déployé manuellement à partir de l'environnement de dev

Nous allons travailler étape par étape pour réaliser ce pipeline.

Première étape – Création Projet Azure DevOps

La première étape consiste à placer les sources du site des exoplanètes dans un repository GitHub.

1. Récupérez sur MooVin les sources du site des exoplanètes
    a. Prenez la version du site des exoplanètes de cette séance, les fichiers ont été quelque peu modifiés pour PostgreSQL
2. Créez un projet Azure DevOps
    a. https://dev.azure.com
    b. Démarrez gratuitement
    c. Créez un projet **privé** avec Git comme outil de version
3. Ajoutez au dépot Azure DevOps Repos le code source

Deuxième étape – Env Dev / Prod App Service

1. Allez sur le site https://portal.azure.com
2. Créez une App Service pour l'environnement de développement
    a. **exoapp-xxx-dev**
    b. **Node 16 LTS**
    c. **Linux**
    d. **West Europe ou France Central**
    e. **Tarification F1 (gratuit)**
3. Créez une App Service pour l'environnement de production
    a. **exoapp-xxx-prod**
    b. **Node 16 LTS**
    c. **Linux**
    d. **West Europe ou France Central**
    e. **Tarification F1 (gratuit)**

Troisième étape – un premier pipeline simple sans DB

1. Dans le projet Azure DevOps, cliquez sur **pipelines** et créez un pipeline
    a. Source Code -> Azure Repo
    b. Configure -> Node.js avec Linux Web App

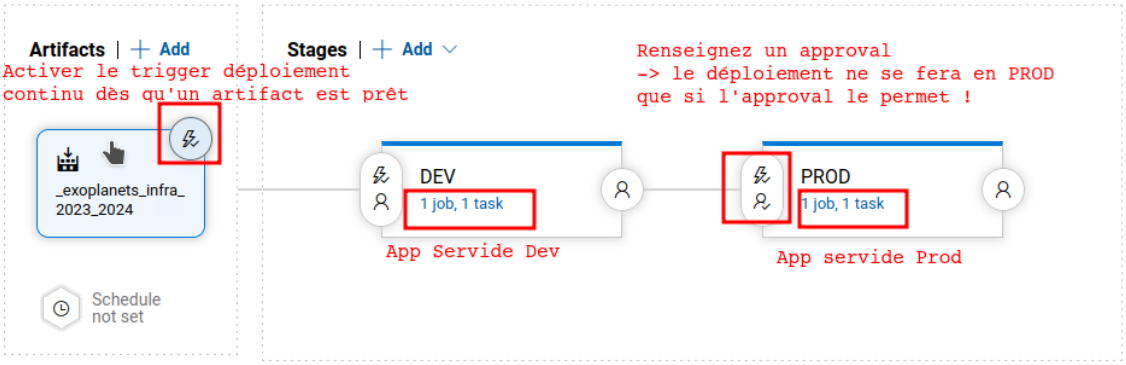
Node.js Express Web App to Linux on Azure
Build a Node.js Express app and deploy it to Azure as a Linux web app.

    c. Renseignez votre App Service **exoapp-xxx-dev**
2. Tester le pipeline et comprenez son fonctionnement
    a. Regardez le fichier azure-pipelines.yml dans votre dépôt
    b. Regardez aussi les jobs / logs dans le menu pipelines
    c. Retrouvez l'exécution du simple test
    d. Vérifiez que l'application est bien déployée sur votre App Service
        i. Le page d'accueil du site devrait être fonctionnel. Le lien vers les exoplanètes nécessite une DB et n'est pas fonctionnel pour l'instant

Quatrième étape – un premier pipeline plus élaboré sans DB

Nous allons séparer le pipeline en 2 pipelines : CI (Continuous Integration) et CD (Continuous Deployment).

1. Editez votre pipeline actuel
    a. Supprimez tout le "stage" Deploy
    b. Ce pipeline sera le pipeline CI responsable de :
        i. Exécuter les tests
        ii. Produire un artifact (un build déployable)
2. Créez le pipeline CD
    a. Activer les **Releases Pipelines**
        i. Cliquez sur Azure Devops (en haut à gauche)
        ii. Organization Settings
        iii. Pipeline Settings
        iv. Décocher Disable creation of classic release pipelines
    b. Retournez dans votre projet Azure DevOps
    c. Sous pipeline, cliquez sur **Release**
    d. Créer un release pipeline
        i. Ajoutez l'artifact produit par le pipeline CI
        ii. Stage 1 devient l'environnement DEV (développement)
        iii. Liez DEV à votre App Service **exoapp-xxx-dev**
        iv. Créez un environnement prod et liez-le à votre app **exoapp-xxx-prod**
        v. Renseignez votre compte vinci comme approval dans l'environnement de production



3. Testez vos pipelines
    a. Faites un simple commit de code
    b. Ce simple commit devrait lancer l'exécution complète des pipelines jusqu'au déploiement de l'application dans l'environnement de développement.
    c. Débogage
        i. Vous avez des logs dans Azure DevOps lorsque les tâches s'exécutent
        ii. Vous avez "Flux des journaux" dans votre App Service

Cinquième étape – Base de données ElephantSQL

1. Allez sur le site ElephantSQL et connectez-vous avec Github
2. Créez une base de données **exoapp-xxx-db-dev** (nouvelle instance)
    a. Prenez la formule Tiny Turtle (gratuite)
    b. Région : Ireland
    c. Exécutez le fichier SQL « create.sql » pour créer la table des exoplanètes présent dans les ressources
3. Créez une base de données **exoapp-xxx-db-prod** (nouvelle instance) de la même manière
4. Utilisation d'une variable d'environnement pour la chaîne de connexion DB
    a. Récupérez la ConnectionString (URL) sur ElephantSQL
    b. Ex : postgres://rlehidcs:qbNtrZhdQbrQvbscLmpAfrrTNLvHmHT@lucky.db.elephantsql.com/rlehidas
    c. Créez une variable d'environnement dans l'App Service exoapp-xxx-dev
        i. Configuration→ Paramètres de l'application
        ii. La variable sera nommée PG_CONN
        iii. Sa valeur sera la ConnectionString d'ElephantSQL
        iv. En image :

5. Modifiez le code (config.js) pour qu'il utilise la variable d'environnement PG_CONN
    a. Dans le code, vous pouvez accéder aux variables d'environnement via **process.env.VARNAME**
    b. Pourquoi utilisez une variable d'environnement ?
6. Testez votre pipeline
    a. Vous devez voir l'application exoplanètes avec le lien vers les exoplanètes fonctionnel !
    b. Débogage
        i. Vous avez des logs dans Azure DevOps lorsque les tâches s'exécutent
        ii. Vous avez "Flux des journaux" dans votre App Service