EXAMEN DE VALIDATION

TaskMaster & Workflow DevOps

Formation : Le Trépied du Développeur

Python Git Docker

Durée estimée : 2h00

Objectif : Valider la capacité à isoler un environnement, gérer un cycle de vie Git complet et livrer une application conteneurisée

AfriAI Solutions

Transformons vos idées en solutions

www.afriaisolutions.com afriaisolution@gmail.com

Mise en Situation

Vous récupérez le code source « **brut** » d'une application de gestion de tâches (**TaskMaster**). Actuellement, ce code n'est pas professionnel : il n'a pas de gestion de dépendances, pas d'historique de version et ne peut pas être déployé facilement.

Votre Mission

« **Professionnaliser** » cette application en appliquant les bonnes pratiques vues en cours : **Isolation**, **Versionning avancé** et **Dockerisation**.

🗁 ÉTAPE 0 : Le Fichier Source

1. Créez un dossier nommé `Examen_VotrePrenom`
2. À l'intérieur, créez un fichier `app.py` et copiez-y le code suivant :

```python
from flask import Flask, jsonify, request
import sqlite3
import os

app = Flask(__name__)
DB_FILE = "tasks.db"

def init_db():
    conn = sqlite3.connect(DB_FILE)
    cursor = conn.cursor()
    cursor.execute('''CREATE TABLE IF NOT EXISTS tasks
                      (id INTEGER PRIMARY KEY AUTOINCREMENT,
                       title TEXT)''')
    conn.commit()
    conn.close()

@app.route('/')
def home():
    # C'est cette ligne qu'il faudra modifier a la fin de l'exercice Git
    return "<h1>TaskMaster v1.0</h1><p>L'application tourne dans Docker
        !</p>"

@app.route('/tasks', methods=['GET'])
def get_tasks():
    conn = sqlite3.connect(DB_FILE)
    cursor = conn.cursor()
    cursor.execute('SELECT * FROM tasks')
    tasks = [{"id": row[0], "title": row[1]} for row in
        cursor.fetchall()]
    conn.close()
    return jsonify(tasks)

if __name__ == '__main__':
    init_db()
    print("Serveur lance sur le port 5000...")
    app.run(host='0.0.0.0', port=5000)
```

Listing 1 – **Code source de TaskMaster** — app.py

MISSION 1 : L'ISOLATION (PYTHON)

Objectif : Ne jamais polluer le système global.

1. Dans le dossier du projet, créez un environnement virtuel nommé `venv`
2. Activez cet environnement
3. Installez la librairie `flask`
4. Générez le fichier `requirements.txt` contenant la liste des dépendances

Preuve de Réussite : Le dossier contient un fichier `requirements.txt` valide avec Flask listé.

MISSION 2 : LE WORKFLOW GIT

Objectif : Prouver que vous maîtrisez les branches et les fusions.

IMPORTANT : Respectez scrupuleusement l'ordre des étapes ci-dessous.

A. Initialisation

1. Initialisez un dépôt Git
2. **CRITIQUE :** Créez un fichier `.gitignore` pour exclure impérativement :
   - Le dossier `venv/`
   - Le dossier `__pycache__/`
   - Les fichiers database (`*.db`)
3. Faites le **Premier Commit** sur la branche `main` avec les fichiers initiaux
4. Message de commit : `"Initialisation du projet"`

B. Le Développement Parallèle (Branche)

1. Créez une nouvelle branche nommée `documentation` et basculez dessus
2. Créez un fichier `README.md` contenant : `"Examen validé par [Votre Nom]"`
3. Ajoutez et commitez ce fichier sur la branche `documentation`
4. Message de commit : `"Ajout du README"`

C. La Fusion (Merge)

1. Revenez sur la branche `main`
2. Fusionnez (merge) la branche `documentation` dans `main`
3. Supprimez la branche `documentation` (elle est devenue inutile)

D. La Modification Finale

1. Restez sur `main`
2. Modifiez le fichier `app.py` : changez le titre
   `<h1>TaskMaster v1.0</h1>` → `<h1>TaskMaster PRO v2.0</h1>`
3. Commitez cette modification
4. Message de commit : `"Mise à jour du titre v2"`

MISSION 3 : LA LIVRAISON (DOCKER)

Objectif : Rendre l'application déployable partout.

1. Créez un fichier `.dockerignore` (Doit exclure `.git`, `venv`, etc.)
2. Créez un fichier `Dockerfile` optimisé :
   - Image de base : `python:3.9-slim` (ou 3.10)
   - Copie des fichiers nécessaires
   - Installation des dépendances via `requirements.txt`
   - Exposition du port 5000
   - Commande de lancement : `python app.py`
3. Construisez l'image sous le nom `examen-taskmaster`
4. Lancez un conteneur basé sur cette image avec la contrainte suivante :

   L'application doit être accessible sur le **port 8000** de votre machine (Rappel : Le conteneur écoute sur le 5000 à l'intérieur)

LIVRABLES À RENDRE

Vous devez envoyer un dossier compressé (**ZIP**) contenant :

- Votre projet complet (sans le dossier `venv`)
- Une capture d'écran obligatoire

La Capture d'Écran Obligatoire

Ouvrez votre terminal et tapez la commande suivante pour prouver votre historique Git :

```
git log -oneline -graph -all
```

Faites une capture d'écran du résultat et incluez-la dans le ZIP.

GRILLE D'ÉVALUATION

(Réservé au Formateur)

Total : /20 points

Section	Critère	Pts	Vérification
ENVIRONNEMENT (4 pts)			
	Venv créé	1	Le dossier existe localement
	requirements.txt	3	Contient Flask
GIT (10 pts)			
	.gitignore	3	**Bloquant :** Si venv commité → 0/3
	Branche créée	2	Visible dans le graphe
	Fusion (Merge)	3	README présent sur main
	Historique (3 étapes)	2	Init → Readme → Update
DOCKER (6 pts)			
	Dockerfile Correct	2	COPY requirements avant RUN pip
	.dockerignore	1	Présent
	Test Final	3	`docker run -p 8000:5000 im` affiche « **TaskMaster PRO v2.0** »

Bonne chance à tous !

« Le succès est la somme de petits efforts répétés jour après jour. »

+224 620 276 939 | contact@afriaisolutions.com | afriaisolution@gmail.com