**PROJET DE DEVELOPPEMENT DE BASE DE COMPOSANT ET SERVICES WEB**

# Exercice 3 : Application RESTful avec Flask ou django

**Objectif :** Développer une application RESTful en utilisant Flask qui utilise Apache Nifi pour ETL et tests avec Postman.

**Tâche :**

* Utilisez Apache Nifi pour extraire et transformer des données de réservations d'hôtel dans MySQL
* Développez une application Flask pour exposer ces données via une API RESTful.
* Utilisez Postman pour tester les endpoints de l'API et vérifier leur fonctionnalité.
* Documentez les tests Postman et assurez-vous que l'API fonctionne correctement.

**Proposition de Plan de Rédaction du Rapport de Projet**

**Introduction**

1. **GENERALITE**
   1. **Contexte et justification**
   2. **Objectifs du Projet**
   3. **Résultats attendus**
   4. **Méthode d’analyse**
   5. **Groupe de travail**
   6. **Planning des travaux**
   7. **Présentation des technologies utilisées**
      1. **Apache NiFi** : Rôle dans l’ETL
      2. **Flask** : pour le développement de l’API./ Exposition des données via l’API RESTful.
      3. **MySQL** : pour la base de données /Stockage des données.
      4. **Postman** : Outil de test et de validation des endpoints.
2. **Conception et Architecture du Système**
   1. **Architecture du Projet**  
      Description de l'architecture globale du système, mettant en évidence le flux de données depuis l’extraction dans MySQL jusqu'à leur exposition via Flask.
   2. **Schéma d’Architecture**  
      un schéma illustrant les composants du projet : NiFi, base de données MySQL, API Flask, et Postman.
   3. **Configuration et processus ETL avec d’Apache NiFi**
      1. **Présentation d’Apache NiFi et de son Utilisation**  
         Description briève Apache NiFi et comment il est utilisé pour l'extraction, la transformation, et le chargement des données.
      2. **Installation et configuration d’Apache Nifi**
      3. **Configuration du Flux ETL**  
         Explication des étapes de configuration du flux de traitement des données dans NiFi :

* **Extraction** : Utilisation de processeurs pour extraire les données depuis MySQL.
* **Transformation** : Nettoyage et mise en forme des données.
* **Chargement** : Chargement des données transformées dans une base de données ou un format adapté pour Flask.
  1. **Schéma du Flux NiFi**  
     un diagramme représentant le flux de données dans NiFi.

1. **Développement de l'API RESTful avec Flask**
   1. **Présentation de Flask et du Développement d’une API RESTful**Explication du rôle de Flask dans le projet et son importance pour créer des APIs RESTful.
   2. **Création de l’environnement virtuel et installation des dépendances**
   3. **Modélisation de la Base de Données**  
      Description de la structure de la base de données utilisée par Flask pour gérer les données de réservations.
   4. **Structure de l’application : organisation des routes, des modèles de données, etc.**
   5. **Endpoints de l'API**  
      Présentation des différents endpoints développés, incluant les détails suivants pour chaque endpoint :
      1. **URL de l’endpoint.**
      2. **Méthode HTTP utilisée (GET, POST, PUT, DELETE) et leurs fonctionnalités.**
      3. **Paramètres d'entrée et de sortie.**
      4. **Connexion à la base de données : expliquer comment Flask se connecte à la base de données peuplée par Nifi.**
      5. **Exemple de requête et de réponse.**
   6. **Exemple de Code et Structure du Projet**  
      un aperçu du code source de l’application Flask, en mettant en avant la structure des fichiers et les routes principales.
2. **Tests et Validation avec Postman**
   1. **Présentation de Postman et de son Utilisation**  
      Explication comment Postman a été utilisé pour tester et valider les endpoints de l’API.
   2. **Scénarios de Tests Définis**  
      Description des scénarios de tests pour chaque endpoint, y compris :
      1. **Requêtes GET pour la récupération des données.**
      2. **Requêtes POST pour l’ajout de nouvelles données.**
      3. **Requêtes PUT**
      4. **Requêtes DELETE**
      5. **Validation des erreurs (par exemple, gestion des données manquantes ou incorrectes).**
   3. **Résultats des Tests avec Captures d’Écran**  
      Documentation des résultats des tests avec des captures d'écran des requêtes Postman, montrant les réponses obtenues et leur conformité.

**Conclusion**

**Annexes**

* Diagrammes supplémentaires.
* Export de la collection Postman.
* Documentation technique détaillée (code source, configuration NiFi, etc.).