VISIPLUS
digital learning

FEUILLE DE TRAVAIL

Proof of concept

Zantour Wael

Liens des Dépôts Git

- **Dépôt Git du Code :** https://github.com/wa2l-99/MedheadApp_PoC
    - Ce dépôt contient :
        - Le code de la PoC versionné.
        - Le(s) fichier(s) de configuration de la chaîne de build d'intégration continue.
        - Le code des tests implémentés.
        - Un fichier README détaillant l'utilisation du projet.

- **Dépôt Git de l'Architecture :** https://github.com/wa2l-99/MedheadApp_Architecture
    - Ce dépôt contient :
        - Tous les documents liés à l'architecture fournis initialement.
        - Un document de reporting.

Contenu du Fichier ZIP

- **Code de la PoC :** Le fichier ZIP joint contient le code de la PoC dans le dossier « MedHeadApp_POC ».
- **Document de Reporting :** Le fichier ZIP inclut également le document de reporting détaillant l'avancement et les résultats du projet dans le dossier « MedheadApp_Architecture » → « Reporting ».

Les compétences et critères d'évaluation

☞ Prenez connaissance de ce tableau qui vous indique les critères sur lesquels vous serez évalué pour chacune des compétences. Les critères vous indiquent également les points d'attention que vous devez avoir dans le rendu de votre copie.

Compétences évaluées	Critères d'évaluation
C.14. Concevoir une architecture adéquate, à partir des exigences et attributs de qualité en réalisant des diagrammes d'architecture et en les formalisant dans un support technique à destination de l'équipe de développement afin de faciliter son usage, son adoption, sa robustesse et son évolutivité.	- le front-end et le back-end sont fonctionnels dans le respect des exigences du développement de la preuve de concept -
C.16. Implémenter un logiciel de qualité, en choisissant des structures de données adaptées et des algorithmes pertinents afin d'assurer la robustesse du logiciel.	- les interactions entre le front et le back sont sécurisées - les tests vérifient la capacité du back-end à assurer une montée en charge
C.17. Tester le logiciel et l'application à plusieurs niveaux en utilisant les méthodologies de test éprouvées afin de garantir la conformité du logiciel au regard des spécifications et la non-régression des fonctionnalités déjà développées.	- un pipeline CI/CD est intégré au repository du code source - des tests vérifient le fonctionnement
C.18. Concevoir une application d'analyse de données massives en intégrant un programme d'apprentissage automatique (machine learning) au développement du logiciel et en utilisant des réseaux de neurones, des algorithmes d'optimisation et de recommandation afin de faire ressortir les tendances utilisateurs.	- les résultats de la PoC fournissent des métriques quant au temps d'exécution de l'API

VISIPLUS
digital learning

Instructions

Intégrer les fonctionnalités par spécialités :

◆ AIDE ◆

Les spécialités existantes sont disponibles : https://digital.nhs.uk/data-and-information/publications/statistical/nhs-workforce-statistics/nhs-workforce-statistics---january-2018

L'architecture TOGAF (ADM) :

◆ AIDE ◆

Les différentes phases : https://conexiam.com/fr/togaf-adm-phases-explained/

Une documentation plus détaillée : https://pubs.opengroup.org/architecture/togaf8-doc/arch/chap03.html

Utiliser le CI/CD de Github directement :

https://docs.github.com/fr/codespaces/setting-up-your-project-for-codespaces/adding-a-dev-container-configuration/setting-up-your-java-project-for-codespaces