|  |
| --- |
| **BTS Services informatiques aux organisations SESSION 2024**  **ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle (recto)**  **Épreuve E5 - Conception et développement d’applications (option SLAM)** |

|  |  |  |  |
| --- | --- | --- | --- |
| **DESCRIPTION D’UNE RÉALISATION PROFESSIONNELLE** | | | **N° réalisation : 1** |
| **Nom, prénom : REICHHART Damien** | | **N° candidat :** **02149931394** | |
| **Épreuve ponctuelle** | **Contrôle en cours de formation X** | **Date :** 28 / 04 / 2025 | |
| **Organisation support de la réalisation professionnelle**  Du code exécuté sur le système d'exploitation d'un serveur. | | | |
| **Intitulé de la réalisation professionnelle**  Atelier de professionnalisation 3 | | | |
| **Période de réalisation :** Septembre 2024 – Janvier 2025 **Lieu :** Cci Campus Alsace Strasbourg  **Modalité :  Seul(e) X En équipe** | | | |
| **Compétences travaillées**  XConcevoir et développer une solution applicative  X Assurer la maintenance corrective ou évolutive d’une solution applicative  X Gérer les données | | | |
| **Conditions de réalisation[[1]](#footnote-1) (ressources fournies, résultats attendus)**  Lister, démarrer, arrêter, modifier et supprimer des conteneurs. Gérer les images Docker. Visualiser les logs et les statistiques des conteneurs. Les résultats attendus comprennent une application web intuitive, robuste et sécurisée, capable de communiquer efficacement avec l’API Docker, ainsi qu’une documentation technique détaillant l’architecture, l’installation et l’utilisation de la solution. | | | |
| **Description des ressources documentaires, matérielles et logicielles utilisées[[2]](#footnote-2)**  **Ressources documentaires :**   * Cahier des charges et spécifications fonctionnelles fournies   Ressources matérielles :   * Poste de travail avec configuration adaptée pour le développement et les tests   **Ressources logicielles :**   * **Environnement de développement intégré (EDI) :** Visual Studio Code, PhpStorm * **Langages et frameworks :**   + PHP   + SQL   + Twig * **Technologies et outils :**   + Docker   + Systèmes de gestion de base de données Mariadb   + Outils de gestion de versions Git et hébergement sur GitLab   + Serveur web pour déploiement Apache   + Mermaid et PlantUML pour la documentation | | | |
| **Modalités d’accès aux productions[[3]](#footnote-3) et à leur documentation[[4]](#footnote-4)**  L’accès aux productions et documentations se font sur gitlab à l’url suivante : <https://gitlab.com/DamienReichhart/CCI-BTS-SIO-23-25-Atelier-Professionnalisation-3>  La documentation du projet est également accessible à cette url : <https://damien-reichhart.fr/> | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs**  Table des matières  [Cahier des charges fonctionnel. 3](#_Toc196467957)  [Formalisme 3](#_Toc196467958)  [Gestion de projet 3](#_Toc196467959)  [Réalisations 4](#_Toc196467960)  [Architecture 5](#_Toc196467961)  [Documentation technique 6](#_Toc196467962)  [Diagrammes de classe 6](#_Toc196467963)  [Diagramme de contexte statique 6](#_Toc196467964)  [Diagramme de cas d’utilisation 7](#_Toc196467965)  [Modèle entité association 7](#_Toc196467966)  [Diagrammes de séquence 8](#_Toc196467967)  [Diagramme de séquence des réseaux docker 8](#_Toc196467968)  [Diagramme de séquence des requêtes http 9](#_Toc196467969)  [Diagramme des volumes docker 9](#_Toc196467970)  [Diagramme du lancement de l’application 10](#_Toc196467971)  [Diagramme des couches de sécurité docker 11](#_Toc196467972)  [Cycle de vie de la requête http 12](#_Toc196467973)  [Intégration docker dans php 13](#_Toc196467974)  [Diagramme des flux de données 14](#_Toc196467975)  [Digramme des étapes des middlewares 15](#_Toc196467976)  [Diagramme de représentation du MVC 16](#_Toc196467977)  [Captures d’écrans de l’application 17](#_Toc196467978)  [Page de connexion 17](#_Toc196467979)  [Gestion d’un conteneur 18](#_Toc196467980)  [Gestion d’un serveur 19](#_Toc196467981)  [Gestion des serveurs 20](#_Toc196467982)  [Gestion des dockerfiles 21](#_Toc196467983)  [Gestion des utilisateurs 23](#_Toc196467984) Cahier des charges fonctionnel. Une application web intuitive, robuste et sécurisée pour docker, capable de communiquer efficacement avec l’API Docker, ainsi qu’une documentation technique détaillant l’architecture, l’installation et l’utilisation de la solution.  Les besoins fonctionnels de la solution ont été définit comme tel :  - Gestion du conteneur  - Possibilité de lancer des conteneurs  - Possibilité de stopper des conteneurs  - Possibilité de modifier/supprimer/ajouter des attribues aux conteneurs  - Possibilité de visualiser les conteneurs en cours / arrêtés / en erreur  - Possibilité de supprimer un conteneur  - Possibilité de modifier un conteneur  - Possibilité de gérer les comptes / utilisateur pour le super admin  - Possibilité d’ajouter / modifier des dockerfiles personnalisés été destinés à la solution  - Possibilité de visualiser le log du conteneur  - Possibilité d’ajouter un server a manager Formalisme Pour ce projet, un formalisme des commits et des tickets a été mis en place afin de garantir un suivi de projet de qualité.  Pour cela, le formalisme suivant a été défini avec les préfixes suivant :   |  |  | | --- | --- | | [FIX] | Définit une correction dans un message de commit | | [FEATURE] | Définit que le ticket représente une fonctionnalité ou que le commit représente l’ajout de  Fonctionnalité | | [RELEASE] | Définit que le commit représente une release | | [DOC] | Définit que le ticket ou le commit représente un ajout ou une modification de documentation | | [HOTFIX] | Définit que le ticket ou le commit a réalisé une correction de bug rapide et urgente sur une des branches de production ( preprod / main ) |  Gestion de projet Le projet a été mené selon la méthodologie Waterfall, ou cascade en français.  Dans ce cadre, l’ensemble des tâches et des étapes du projet ont été planifiées en amont, puis structurées sous forme de tickets et de jalons (Milestones).  Chaque membre de l’équipe s’est vu attribuer des tickets correspondant aux fonctionnalités à développer ou aux bugs à corriger. Cela permettait un suivi clair de l’avancement, une activitée en cours était facilement identifiable grâce aux tickets assignés, assurant ainsi une bonne coordination au sein de l’équipe. Réalisations Dans la réalisation de ce projet, j’ai eu la chance de réaliser les éléments suivants :  - Création du schéma de la base de données.  - Gestion des interactions avec les autres serveurs via un client ssh.  - Gestion des interactions avec docker en temps réel dont la mise en place de l’architecture qui sera définit ci-dessous. - Gestion des conteneurs :  - Ajout / suppression / modification des conteneurs  - Ajout / suppression / modification des attributs des conteneurs avec le log, stats, ports, volumes, environnement et commandes.  - Gestion du statut marche / arrêt d’un conteneur  - Gestion des dockerfiles personnalisés pour la solution, de l’étape de création / modification jusqu’à l’envoi et l’utilisation sur le serveur.  - Mise en place du système d’authentification  - Amélioration de la superclasse Model pour permettre des interactions plus simples avec la base de données  - Rédaction de la documentation liée aux diagrammes de classe, du readme et de la documentation sur l’environnement de développement.  - Réalisation de l’intégralité des templates twig  - Réalisations de scripts CI/CD  - Rédaction du cahier des charges avec mon coéquipier.  - Réalisation du stack de déploiement docker permettant d’avoir un environnement de développement et de production fiable et déplorable facilement.  - Rédaction des utilitaires de projet (Makefile, etc…)  - Rédaction de la documentation :  - Documentation administrateur  - Documentation utilisateur  - Diagramme de cas d’utilsation  - Diagramme de contexte static  - Modèle entité association  - Diagrammes de séquences en tout genre pour expliquer les étapes du fonctionnement avec docker et les servers.  - Diagrammes de classe  - Documentation complète de l’environnement de développement  - Documentation sur les étapes de CI/CD  - Rédaction du guide d’installation Architecture Voici l’architecture des fichiers du projet :  Racine du projet  ├───assets  │ ├───css  │ ├───img  │ └───js  ├───atelierHub  ├───docker  │ ├───apache  │ │ ├───certs  │ │ ├───config  │ │ └───logs  │ ├───db  │ │ ├───AtelierPro  │ │ ├───mysql  │ │ ├───performance\_schema  │ │ └───sys  │ └───php  ├───documentation  │ ├───captures  │ ├───environement  │ ├───gestion\_projet  │ ├───Maquette  │ ├───technique  │ │ └───diagrammes  │ │ ├───classes  │ │ └───sequence  │ └───utilisateur  ├───logs  ├───public  ├───sql  ├───src  │ ├───Controller  │ ├───Entity  │ │ ├───Docker  │ │ └───Form  │ ├───Enum  │ ├───Helper  │ ├───Middleware  │ ├───Model  │ ├───Service  │ └───Structure  ├───template  │ ├───admin  │ ├───auth  │ ├───container  │ ├───docker  │ ├───dockerfile  │ ├───error  │ ├───layout  │ ├───server  │ └───user  └───tests Documentation techniqueDiagrammes de classe Ici vous pourrez trouver le diagramme de classe représentant l’architecture docker :    Ici vous pourrez trouver l’architecture des contrôleurs de l’application :   Diagramme de contexte statique  Diagramme de cas d’utilisation  Modèle entité association  Diagrammes de séquenceDiagramme de séquence des réseaux docker  Diagramme de séquence des requêtes http  Diagramme des volumes docker  Diagramme du lancement de l’application  Diagramme des couches de sécurité docker  Cycle de vie de la requête http  Intégration docker dans php  Diagramme des flux de données  Digramme des étapes des middlewares  Diagramme de représentation du MVC  Captures d’écrans de l’application Vous trouverez ci-dessous des captures d’écran de l’application : Page de connexion  Gestion d’un conteneur      Gestion d’un serveur    Gestion des serveurs  Gestion des dockerfiles        Gestion des utilisateurs |
|  |

1. En référence aux *conditions de réalisation et ressources nécessaires* du bloc « Conception et développement d'applications » prévues dans le référentiel de certification du BTS SIO. [↑](#footnote-ref-1)
2. Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l’annexe II.E du référentiel du BTS SIO. [↑](#footnote-ref-2)
3. Conformément au référentiel du BTS SIO « *Dans tous les cas, les candidats doivent se munir des outils et ressources techniques nécessaires au déroulement de l’épreuve. Ils sont seuls responsables de la disponibilité et de la mise en œuvre de ces outils et ressources. La circulaire nationale d’organisation précise les conditions matérielles de déroulement des interrogations et les pénalités à appliquer aux candidats qui ne se seraient pas munis des éléments nécessaires au déroulement de l’épreuve.* ». Les éléments peuvent être un identifiant, un mot de passe, une adresse réticulaire (URL) d’un espace de stockage et de la présentation de l’organisation du stockage. [↑](#footnote-ref-3)
4. Lien vers la documentation complète, précisant et décrivant, si cela n’a été fait au verso de la fiche, la réalisation professionnelle, par exemples service fourni par la réalisation, interfaces utilisateurs, description des classes ou de la base de données. [↑](#footnote-ref-4)