|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Nom de naissance* |  | LOPEZ |
| *Nom d’usage* |  | Entrez votre nom d’usage ici. |
| *Prénom* |  | Nicolas |
| *Adresse* |  | 85 Avenue de Gérone, 66000 PERPIGNAN |
|  |  |

|  |  |
| --- | --- |
| **Titre professionnel visé** | |
|  | |
| Concepteur développeur d’applications | |
|  | |
| **Modalité d’accès :** | |
|  | |
|  | Parcours de formation |
|  | Validation des Acquis de l’Expérience (VAE) |
|  | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Présentation du dossier** | | | |
|  | | | |
|  | | | |
|  | | | |
|  | Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel. **Ce titre est délivré par le Ministère chargé de l’emploi.**  Le DP appartient au candidat. Il le conserve, l’actualise durant son parcours et le présente **obligatoirement à chaque session d’examen**.  Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.  Il est consulté par le jury au moment de la session d’examen. | |  |
|  | | | |
| **Pour prendre sa décision, le jury dispose :** | | | |
| 1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l’entretien professionnel ou de l’entretien technique ou du questionnement à partir de productions. 2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle 3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d’un parcours de formation 4. de l’entretien final (dans le cadre de la session titre).   *[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels*  *du ministère chargé de l’Emploi]* | | | |
| **Ce dossier comporte :** | | | |
| * pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ; * un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d’un titre, d’un diplôme, d’un certificat de qualification professionnelle (CQP) ou des attestations de formation ; * une déclaration sur l’honneur à compléter et à signer ; * des documents illustrant la pratique professionnelle du candidat (facultatif) * des annexes, si nécessaire. | | | |
| *Pour compléter ce dossier, le candidat dispose d’un site web en accès libre sur le site.* | | | |
|  | | [**http://travail-emploi.gouv.fr/titres-professionnels**](http://travail-emploi.gouv.fr/titres-professionnels) | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sommaire** | | | |
| **Exemples de pratique professionnelle** | | | |
| **Développer une application sécurisée** | | **p.** | **5** |
|  |  Installer et configurer son environnement de travail en fonction du projet p. | p. | 5 |
|  |  Développer des interfaces utilisateur p. | p. | 7 |
|  |  Développer des composants métier p | p. | 9 |
|  |  Contribuer à la gestion d’un projet informatique p | p. | 11 |
|  |  |  |  |
| **Concevoir et développer une application sécurisée organisée en couches** | | **p.** | **13** |
|  |  Analyser les besoins et maquetter une application p. | p. | 13 |
|  |  Définir l’architecture logicielle d’une application p. | p. | 15 |
|  |  Concevoir et mettre en place une base de données relationnelle p | p. | 18 |
|  |  Développer des composants d’accès aux données SQL et NoSQL p | p. | 20 |
|  |  |  |  |
| **Préparer le déploiement d’une application sécurisée** | | **p.** | **22** |
|  |  Préparer et exécuter les plans de tests d’une application p. | p. | 22 |
|  |  Préparer et documenter le déploiement d’une application p. | p. | 24 |
|  |  Contribuer à la mise en production dans une démarche DevOps p | p. | 26 |
|  |  |  |  |
|  |  |  |  |
| **Titres, diplômes, CQP, attestations de formation** *(facultatif)* | | p. |  |
| **Déclaration sur l’honneur** | | p. |  |
| **Documents illustrant la pratique professionnelle** *(facultatif)* | | p. |  |
| **Annexes** *(Si le RC le prévoit)* | | p. |  |

**Exemples de pratique**

**professionnelle**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **1** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°1***  | | | | ***Installer et configurer son environnement de travail en fonction du projet*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Au démarrage de mon alternance chez REVIMPORT S.A.S, j’ai pris en charge la création et la configuration complète de l’environnement de développement nécessaire pour concevoir des extensions en langage AL pour l’ERP Microsoft Business Central (SaaS).  Concrètement, j’ai :   * Installé Visual Studio Code et configuré les extensions AL Language et AL Test Runner pour faciliter l’écriture et le débogage de code. * Créé un sandbox Business Central afin de tester les futures extensions AL avant le déploiement sur la production, elle est disponible via le service SaaS donc pas besoin de docker ici. * Intégré un dépôt Git pour versionner mes sources. * Vérifié la connectivité, exécuté des déploiements d’essai et validé le bon fonctionnement des objets AL (pages, tables, codeunits) directement dans le client Web de Business Central. * Commencer la documentation du projet de migration. | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Voici les différents moyens et outils mobilisés   * IDE & extensions : Visual Studio Code + AL Language, AL Test Runner, GitLens * Système d’exploitation : Windows 11 Pro * ERP : Business Central SaaS (Azure Sandbox) * Ancien ERP : Navision 2009 Classic * Contrôle de version : Git * Automatisation : PowerShell * Documentation : Readme et guidelines AL | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| Tuteur: Carlos CURTO (Responsable informatique) | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***REVIMPORT S.A.S*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **Service Informatique** | | | | | |
| **Période d’exercice** | |  | **Du** | | **31/08/2024** | | **au** | **16/09/2024** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| * Rédaction d’un guide pas-à-pas d’installation et de configuration à usage interne. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **1** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°2***  | | | | ***Développer des interfaces utilisateur*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Dans le cadre du projet Kigo réalisé au centre de formation IDEM, j’ai conçu, prototypé et implémentél’interface utilisateur en respectant l’ensemble des critères suivants :   * Conformité au dossier de conception   + Rédaction du cahier des charges, concepts évoqués et dictionnaire de données. **Annexe 01**   + Réalisation de wireframes et d’une maquette sur Figma. **Annexe 02** * Responsive design & adaptation multi-supports   + Implémentation d’un layout responsive avec Tailwind CSS : grilles fluides, classes sm:, md:, lg: pour mobiles, tablettes et desktop. **Annexe 03**   + Tests manuels sur émulateurs (Chrome DevTools) et smartphones réels pour valider l’affichage et l’ergonomie selon la taille, le type et la disposition du support. * Respect de la charte graphique   + Application de la configuration Tailwind. **Annexe 04**   + Utilisation des couleurs, typographies et animations définies pour tous les composants (boutons, formulaires, cartes). * Conformité à la réglementation en vigueur (RGPD)   + Mise en place d’une bannière de consentement cookies avant tout dépôt de traceurs.   + Stockage des données personnelles (formulaires de contact) uniquement après consentement explicite, et purge automatique des logs au-delà de 13 mois. * Réalisation de tests unitaires pour les composants concernés   + Création d’un test unitaire pour le composant PrimaryButton avec Jest et React Testing Library : | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| * Framework & bundler : React 18 + Vite * Langage & styles : TypeScript, Tailwind CSS, PostCSS * Prototypage : Figma (wireframes, maquettes haute-fidélité) * Versioning : Git & GitHub (branches feature / main, pull requests) * Tests : Jest, React Testing Library * IDE : Visual Studio Code | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| En autonomie partielle tout en bénéficiant de l’encadrement pédagogique des formateurs de l’IDEM. Une partie de la conception a été réalisée en groupe de classe.  Revue de maquette avec les formateurs de l’IDEM | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | Cliquez ici pour taper du texte. | | | | | |
| **Période d’exercice** | |  | **Du** | | **15/04/2024** | | **au** | **17/06/2024** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **1** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°3***  | | | | ***Développer des composants métier*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Au sein du projet de migration navision > Business central chez REVIMPORT S.A.S, pendant l’alternance, j’ai conçu et implémenté une extension monolithique regroupant tables, pages, codeunits, rapports, permissions et enums afin de limiter les dépendances multiples.  Concrétement, j’ai :   * **Conception modulaire et POO :**   Décomposé la logique métier en plusieurs codeunits et respecté le principe de responsabilité  Unique (SRP) : Chaque procédure gère un seul aspect métier, et les données sont passées en paramètre plutôt qu’accédées de façon globale. **Annexe 05**   * **Sécurisation serveur :**   Validation systématique des entrées, gestion des droits via PermissionSet dédiés et journalisation des erreurs métier et levée d’exceptions contrôlées pour garantir l’intégrité des traitements. **Annexe 06**   * **Convention de nommage :**   Numérotation des objets selon les plages recommandées et utilisation du PascalCase pour les identifiants logiques (CalculerPR, HistoPRTParam ) et préfixe « Revimport » sur les tables spécifiques à l’entreprise.  **Tables/tableextensions : 50 000–50 999**  **Pages/pageextensions : 51 000–51 999**  **Codeunits/permissions/enums : 52 000–52 999**   * **Documentation du code :**   En-tête détaillé avec historique de versions et référence aux spécifications NSC (ex. NSC1.12, NSC2.01, MIGRATION\_400).  Documentation code source en **Annexe 07**  Commentaires de procédure décrivant l’objectif et les paramètres et plan de mise à jour de la documentation (/// <summary>…) pour un affichage dans IntelliSense.  /\*  CalculerPRT : calcule le Prix de Revient Théorique  @param HistoPRTParam – record HistoriquePRTTable à mettre à jour  \*/  procedure CalculerPRT(VAR HistoPRTParam: Record "HistoriquePRTTable")   * **Test de sécurité :**   Intégration d’AL Code Cop pour détecter les appels non sécurisés et les mauvaises pratiques.  Analyse régulière pour identifier les vulnérabilités et corriger les règles OWASP pertinentes.  (Open Worldwide Application Security Project) | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Langage & IDE : AL Language sur Visual Studio Code  ERP & sandbox : Business Central SaaS (Azure Sandbox)  Versioning & CI/CD : GitHub (branches feature/main), Azure DevOps pipelines  Tests : AL Test Runner  Analyse statique & sécurité : AL Code Cop, OWASP ZAP (audit ponctuel) | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| Tuteur: Carlos CURTO (Responsable informatique) | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | **Cliquez ici pour taper du texte.** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **REVIMPORT S.A.S** | | | | | |
| **Période d’exercice** | |  | **Du** | | **31/08/2024** | | **au** | **15/03/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Rédaction d’un Wiki interne détaillant l’architecture des codeunits et les patterns POO appliqués.  Proposition d’un plan d’amélioration pour renforcer la documentation XML et automatiser davantage la génération de rapports de sécurité.  Mise en place d’une revue de code bi-hebdomadaire pour maintenir la qualité et la conformité aux standards. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **1** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°4***  | | | | ***Contribuer à la gestion d’un projet informatique*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| **Planification des tâches selon de délai défini :**  A chaque début de semaine, mon tuteur met à disposition une liste ordonnée de migrations et d’opérations à réaliser (création de tables, déploiement d’extensions, correction de bugs).  Chaque tâche est décrite sur une ligne avec date cible, responsable, et case à cocher pour « En cours » , « Testé », « Validé ».  Avant toute mise en production, un statut « Test end-to-end » doit être coché dans OneNote pour attester du passage en revue fonctionnelle.  **Suivi rapproché et gestion des retards :**  Chaque matin, je participe à une réunion avec mon tuteur pour **ajuster les priorités de la journée** (bugs critiques, montées de version, patchs de sécurité, etc.). Pour la communication interne, j'utilise **OneNote et Outlook**. Des **alertes** sont configurées pour toute échéance dépassée afin de garantir une visibilité constante sur l'avancement des tâches.  **Mise en œuvre des procédures qualité :**  J'ai mis en place des contrôles qualité rigoureux au sein de mon pipeline de développement. Cela inclut l'analyse statique via AL Code Cop intégré dans Azure DevOps CI, qui bloque les montées en version en cas d'avertissements critiques. Je maintiens également des check-lists de revue complètes dans OneNote, alignées sur les recommandations de Microsoft concernant les conventions de nommage AL, les permissions, les bonnes pratiques POO et la sécurisation des endpoints. De plus, j'assure la conformité aux standards en référençant directement la documentation officielle de Business Central dans OneNote.  **Environnement et architecture projet :**  J'utilise une Sandbox Cloud Business Central comme environnement de pré-production, répliquant la base de données de production pour homogénéiser les jeux de données et les configurations.  **Outils collaboratifs selon la méthode de développement :**  J'applique une méthode Agile légère avec des sprints d'une semaine et des livrables intermédiaires validés par mon tuteur et les utilisateurs clés. Pour cela, j'utilise divers outils : OneNote et SharePoint pour une documentation vivante, la liste des livrables et le suivi de l'avancement et GitHub pour le versionnement du code AL (branches de fonctionnalités, pull requests, revues de code). | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| **Planification & documentation :** OneNote (partagé via SharePoint) pour lister les tâches, décrire les procédures qualité, héberger les check-lists et les diagrammes d’architecture.  **Communication & suivi :** Outlook pour l’envoi d’e-mails d’alerte, Teams/visioconférence pour les points quotidiens, drapeaux et cases à cocher dans OneNote pour repérer les retards.  **Contrôle de version & CI/CD :** GitHub (branches feature/main, pull requests) et Azure DevOps pipelines (déploiement automatique vers sandbox, exécution d’AL Code Cop).  **Environnement de développement :** Sandbox Business Central SaaS (pré-production) reproduisant la base de données de production, provisionné via scripts PowerShell et Docker Compose documentés dans OneNote. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| Tuteur: Carlos CURTO (Responsable informatique) | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***REVIMPORT S.A.S*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | Cliquez ici pour taper du texte. | | | | | |
| **Période d’exercice** | |  | **Du** | | **31/08/2024** | | **au** | **15/03/2024** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **2** | | | **Concevoir et développer une application sécurisée organisée en couches** | | | | |
| ***Exemple n°1***  | | | | ***Analyser les besoins et maquetter une application*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| **Recueil et validation des besoins :**  J'ai commencé par étudier le cahier des charges Kigo pour identifier les cas d'usage, tels que la création de projets, les candidatures, le fil d'actualité, les profils utilisateurs et les notifications. Ensuite, j'ai rédigé une liste des exigences fonctionnelles et techniques, couvrant l'ensemble des besoins exprimés sous forme de user stories et d'un dictionnaire de données. **Annexe 01**  **Maquettage, prototypage et parcours utilisateur :**  J'ai élaboré un zoning global de l'application sur Figma, définissant les zones fixes (header, menu latéral, fil d'actualité) et dynamiques (listes, formulaires). Ensuite, j'ai réalisé des wireframes basse fidélité pour chaque écran clé (dashboard, page projet, profil, post, message d'information). J'ai aussi créé des maquettes haute-fidélité en français, avec une terminologie claire et simple (niveau B1 CECRL). Enfin, j'ai monté un prototype cliquable dans Figma, simulant la navigation réelle entre les écrans.  **Structuration du dossier de conception : Annexe 08 & Annexe 09 & Annexe 10**   * Analyse des besoins * Persona * User journey map * Etude concurence * Conception fonctionnelle (wireframes, zoning) * Conception UI (maquettes, prototype) * Spécifications techniques (couches, API, sécurité) * Recette et tests | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Notion pour la planification et la documentation des tâches, procédure qualité et chech-lists.  GitHub pour le contrôle de versions .  VsCode, Figma pour les maquettes et Word pour la cahier des charges. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| En autonomie partielle tout en bénéficiant de l’encadrement pédagogique des formateurs de l’IDEM. Une partie de la conception a été réalisée en groupe de classe.  Revue de maquette avec les formateurs de l’IDEM | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **CDA – Concepteur Développeur Applications** | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/02/2025** | | **au** | **30/04/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Les exports des maquettes, wireframes et diagrammes sonts annexés en images pour illustrer chaque point (zoning, prototype, flow-chart). Annexe 08 et 09 et 10 | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **2** | | | **Concevoir et développer une application sécurisée organisée en couches** | | | | |
| ***Exemple n°2***  | | | | ***Définir l’architecture logicielle d’une application*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| **Architecture multicouche et rôle de chaque strate :**  Pour Kigo, j’ai retenu une architecture en 5 couches clairement délimitée, suivant les bonnes pratiques d’un système réparti et sécurisé :  Couche Présentation –FrontEnd :  J'ai travaillé sur le front-end en utilisant React, Redux, TypeScript, Tailwind et Vite. Mes responsabilités incluaient l'affichage des écrans (login, dashboard, profil, projets, posts), la gestion du routage avec react-router-dom, l'orchestration du state global et le déclenchement des appels API.  Côté sécurité, j'ai mis en place la sanitisation des entrées utilisateurs, un contrôle d'accès côté client (redirection des non-authentifiés via AuthContext), et une protection contre le XSS grâce à des composants isolés et l'utilisation judicieuse des attributs.  Couche API / Contrôleurs :  J'ai développé le back-end en utilisant Symfony et API Platform. Ma responsabilité était d'exposer les ressources métier via des endpoints REST/GraphQL, d'orchestrer la validation des requêtes et de convertir les données entre JSON et les entités Doctrine.  En matière de sécurité, j'ai mis en place un CORS strict via NelmioCorsBundle, géré l'authentification JWT (ou les sessions Symfony), vérifié les permissions avec IsGranted et @Security dans les contrôleurs, et configuré le throttling et le firewall Symfony.  Couche Métier ( Services & Logique ) :  J'ai structuré le back-end avec des services Symfony et objets Doctrine, en adoptant une approche Code-first via API Platform. Ma responsabilité était d'encapsuler la logique métier (gestion des projets, candidatures, notifications, recommandations) et de séparer la logique complexe (calcul de compatibilité des compétences, envoi de mails) des opérations CRUD basiques.  Concernant la sécurité, j'ai mis en œuvre l'injection de dépendances sécurisée, une validation stricte des DTO avant persistance, ainsi que le chiffrement ou le hachage des données sensibles (mots de passe, tokens).  Couche Persistance ( Données )  J'ai géré la base de données en utilisant Doctrine ORM avec MySQL. Ma responsabilité était de modéliser et stocker les entités (User, Profile, Project, Post, Message, Media, Contact, TypeContact), d'exécuter les transactions et de garantir l'intégrité référentielle.  Côté sécurité, j'ai mis en place le chiffrement SSL/TLS pour les données en transit, des permissions pour l'utilisateur de connexion, ainsi que des backups chiffrés avec une politique de rétention.  Couche infrastructure & Déploiement  J'ai géré le déploiement et les opérations en utilisant Docker/Lando, Nginx. Mes responsabilités incluaient la containerisation des services, l'orchestration du déploiement (build React, migration Doctrine, fixtures), la surveillance de l'état (healthchecks), et le maintien d'un canal sécurisé (HTTPS, certificats Let's Encrypt).  En matière de sécurité, j'ai mis en place la segmentation réseau (front ↔ API ↔ DB), utilisé des variables d'environnement chiffrées, effectué des scans de vulnérabilités Docker, et appliqué des politiques d'images immuables.  Optimisation Ecologique de l’application  Afin de minimiser l'empreinte écologique de l'application, j'ai mis en œuvre plusieurs stratégies clés :  **Côté front-end**, j'ai optimisé les performances grâce au **tree-shaking et au code splitting** via Vite, assurant que seuls les bundles nécessaires soient chargés. J'ai également activé la **compression GZIP/Brotli** des assets, supprimé les dépendances inutilisées, et mis en place le **cache HTTP** (Cache-Control) ainsi que les service workers PWA pour réduire les requêtes réseau.  **Pour le back-end**, j'ai veillé à l'efficacité des **requêtes Doctrine** (jointures conditionnelles, pagination) et intégré une **mise en cache Redis** pour les données lourdes (recommandations, profils publics). J'ai aussi optimisé la consommation des ressources serveur par l'**extinction automatique des containers inactifs** et un **scaling horizontal maîtrisé** basé sur les métriques de charge.  Concernant l'**hébergement et l'infrastructure**, j’aurais privilégié un **fournisseur cloud proposant des clusters bas-carbone** et utilisé des **spot instances ou des réservations à long terme** pour diminuer l'empreinte énergétique globale. | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Pour mes projets, j'utilise Draw.io afin de créer des diagrammes (C4, flow-chart), formalisant la répartition des couches et les flux de données.  La gestion de code et le CI/CD sont assurés via GitHub (avec pull requests) et les pipelines Azure DevOps pour le build front, les tests et le déploiement.  J'utilise Docker et Lando pour la conteneurisation, garantissant un environnement de développement local identique à la production.  Enfin, pour le logging et le monitoring, je m'appuie sur Sentry pour les erreurs front et back, et sur Prometheus/Grafana pour les métriques. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| En autonomie partielle tout en bénéficiant de l’encadrement pédagogique des formateurs de l’IDEM. Une partie de la conception a été réalisée en groupe de classe. | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **Concepteur Développeur Applications** | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/02/2025** | | **au** | **30/04/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **2** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°3***  | | | | ***Concevoir et mettre en place une base de données relationnelle*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Pour cette activité type, j’ai choisi de présenter mon projet de fin d’études BlockLumen.  Pour l'élaboration du **MCD Merise** (**voir Annexe 11**), j'ai veillé à **définir chaque entité** avec une clé primaire unique et des attributs atomiques, sans attribut multivalué. J'ai ensuite **modélisé les associations** avec leurs cardinalités (1 – n, n – m), en qualifiant les liaisons n – m par une table d'association, et j'ai **exclu toute redondance** pour garantir que chaque information n'apparaisse qu'une seule fois, prévenant ainsi les anomalies d'insertion, de mise à jour et de suppression.  Le **MPD Merise** (**voir Annexe 12**) que j'ai élaboré **traduit fidèlement les exigences fonctionnelles** de BlockLumen. Les tables couvrent l'ensemble des cas d'usage (utilisateurs, portefeuille, détentions, transactions, progression pédagogique, etc.), et les types SQL (VARCHAR, DECIMAL, DATETIME, ENUM) sont précisément choisis pour correspondre aux données métier. Les relations physiques (FOREIGN KEY) et les index sont optimisés pour des requêtes performantes (recherche par symbole, jointures portefeuille ↔ transactions).  J'ai scrupuleusement respecté les règles de nommage définies : les tables sont en snake\_case et au singulier (ex. user, wallet\_holding, trade), les colonnes en minuscules avec des underscores (created\_at, average\_price), les clés primaires finissent systématiquement par \_id, et les contraintes sont nommées fk\_<table>\_<référence>. Cette convention uniforme facilite grandement la lecture et l'industrialisation des scripts SQL.  L'intégrité, la sécurité et la confidentialité des données sont assurées par plusieurs mécanismes. La cohérence référentielle est garantie par des FOREIGN KEY … ON DELETE CASCADE, et des contraintes d'unicité (UNIQUE) sont appliquées sur les champs comme username et email pour éviter les doublons. J'ai mis en place le chiffrement des mots de passe en base (password\_hash via bcrypt) avant insertion. L'accès au SGBD est restreint à un utilisateur MySQL dédié avec des droits minimaux, et les connexions TLS sont configurées via Docker-Compose. Enfin, un audit des opérations critiques (INSERT/UPDATE/DELETE) est possible via des triggers ou les logs MySQL.  La base de données de test est créée avec un jeu d'essai complet et peut être restaurée en cas d'incident. Le fichier docker-compose.yml initialise automatiquement une instance MySQL à chaque démarrage et importe le dump BlockLumenBDD.sql (**fixtures voir Annexe 13**). En cas de corruption, il suffit de relancer docker-compose up --force-recreate db pour retrouver l'état initial. Les jeux de données de test couvrent des utilisateurs fictifs, des portefeuilles de cryptos, des historiques de transactions et la progression pédagogique.  La documentation technique des bases de données est compréhensible, en français ou en anglais (niveau B1 CECRL). Le MPD ainsi que le dictionnaire des données (voir Annexe 3) sont rédigés en français clair, et les commentaires SQL dans les scripts de création de tables expliquent le rôle de chaque colonne et contrainte. | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Modélisation Merise : draw.io pour MCD  SGBD & test : MySQL 8 en conteneur Docker (Docker-Compose)  Gestion de versions : GitHub pour les scripts SQL.  Documentation : Markdown et diagrammes Draw.io intégrés au dépôt pour faciliter la lecture. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| En autonomie totale. | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **Concepteur Développeur Applications** | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/03/2025** | | **au** | **20/06/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| **Annexe 11 : MCD**  **Annexe 12 : MPD**  **Annexe 13 : Fixtures** | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **2** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°4***  | | | | ***Développer des composants d’accès aux données SQL et NoSQL*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Dans le cadre de mon projet de fin d'études BlockLumen, j'ai réalisé la conception et l'implémentation des composants d'accès aux données SQL. J'ai créé les entités TypeORM nécessaires (User, Wallet, WalletHolding, Trade, Price, Learn, UserLearn, Preference) ainsi que leurs contrôleurs Express associés (UserController, WalletController, etc.), garantissant toutes les opérations CRUD (voir Annexe 14).  J'ai ensuite mis en place les routes Express pour chaque entité et développé la couche d'initialisation (index.ts) pour injecter les repositories TypeORM. La gestion des exceptions a été une priorité, avec des retours 404 pour les entités introuvables et une gestion robuste des erreurs serveur via des blocs try/catch pour des statuts HTTP 500 (voir Annexe 15).  Afin de fluidifier le développement, j'ai automatisé le démarrage de la base MySQL et du backend via Docker Compose, et configuré le hot-reload. Pour l’intégrité et la confidentialité des données, j'ai intégré un middleware d’authentification/autorisation JWT et des guards d’autorisation pour chaque route.  Par ailleurs, au-delà du cast des paramètres URL avec parseInt(), j’ai défini et appliqué des schémas de validation et de sanitisation des corps de requête (via class-validator/express-validator), et mis en place un rate-limiter pour prévenir les envois malveillants ou excessifs. Enfin, j'ai mis en place des audits de sécurité automatisés en utilisant les règles de sécurité ESLint. | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Technologies : Node.js (v18+), TypeScript, Express.js, TypeORM, MySQL, Docker, JWT, bcrypt.  Outils de développement : VS Code (avec ESLint/Prettier), Postman (tests d’API), Git/GitHub, Nodemon.  Méthodologie : modélisation Merise (MCD, MLD, MPD), écriture de scripts SQL (BlockLumenBDD.sql), documentation (README, dictionnaire de données), versioning Git. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| **En autonomie complète** | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | **Concepteur Développeur Applications** | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/03/2025** | | **au** | **20/06/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **3** | | | **Préparer le déploiement d’une application sécurisée** | | | | |
| ***Exemple n°1***  | | | | ***Préparer et exécuter les plans de tests d’une application*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| **1. Couverture du plan de tests** Je construis un plan de tests qui décrit, pour chaque fonctionnalité retenue dans le cahier des charges — authentification, création et consultation de portefeuilles, simulation d’achats et de ventes, suivi des modules pédagogiques — à la fois le « happy path » et les scénarios d’exception (solde insuffisant, accès non autorisé, données manquantes ou invalides). J’intègre ce plan directement dans ma documentation, sous forme de tableau clair, afin que chaque membre de l’équipe comprenne immédiatement quels comportements sont validés et quelles erreurs sont gérées.  **2. Mise en place d’un environnement de tests isolé** Je définis un fichier docker-compose.test.yml dédié, dans lequel un service MySQL initialise immédiatement la base de données de test à partir de mes fixtures SQL, et un service API exécute les migrations TypeORM avant de lancer les suites de tests. Ainsi, je garantis que la base de données de test ne mélange jamais les données de développement ou de production, et que chaque exécution part d’un état connu.  **3. Exécution conforme au plan de tests** Dans ce même environnement Docker, j’appelle npm test, qui déclenche Jest (avec Supertest pour les appels HTTP) afin de parcourir automatiquement chaque cas défini dans mon plan. Je surveille attentivement la sortie console : chaque test nominal ou d’erreur doit passer, et tout échec signale immédiatement un écart entre le comportement réel et le comportement attendu.  **4. Cohérence des résultats obtenus** À la fin de chaque exécution, je consulte le rapport généré (« Tests: 30 passed, 0 failed ») et j’insère un extrait annoté dans ma documentation de déploiement. En parallèle, je conserve les logs de build et de migrations (via npm run typeorm:migrate et npm run typeorm:revert) pour montrer que, même en cas d’incident, je peux restaurer proprement la base de test à son état initial.  **5. Anticipation des évolutions et des enjeux de sécurité** J’établis une routine de revue trimestrielle de mon plan de tests : lorsque je mets à jour Node.js, TypeORM ou MySQL, je réexamine l’intégralité des scénarios pour y ajouter les nouveaux cas d’erreurs ou les failles de sécurité récemment identifiées (CVE). Je note aussi dans ma documentation que je prévois d’intégrer bientôt des tests de charge et des analyses de vulnérabilité automatisées, afin de rester aligné sur les meilleures pratiques de l’industrie. | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Je m’appuie sur :   * Docker Compose (docker-compose.test.yml) pour isoler les services MySQL et API, et charger mes fixtures SQL automatiquement. * TypeORM pour exécuter les migrations avant chaque session de tests (npm run typeorm:migrate / revert). * Jest avec Supertest pour écrire et lancer les suites de tests d’intégration HTTP couvrant tous les endpoints de l’API. * GitLab CI (ou GitHub Actions) pour automatiser l’exécution des tests à chaque push sur la branche de développement. * Documentation interne (OneNote et Markdown) pour centraliser le plan de tests, les extraits de rapports et les procédures de rollback. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| **Seul en autonomie** | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | Cliquez ici pour taper du texte. | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/03/2025** | | **au** | **20/06/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **3** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°2***  | | | | ***Préparer et documenter le déploiement d’une application*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| J’ai commencé par formaliser chaque étape du déploiement dans un guide unique, depuis la préparation des images Docker jusqu’à l’activation des services en production. Cette procédure décrit précisément la configuration des variables d’environnement, la génération des certificats TLS, la mise à jour des secrets et la migration de la base de données.  Pour chaque phase, j’ai développé des scripts Bash: un script build.sh qui compile l’image API et génère les artefacts React, un script deploy.sh qui pousse les images vers notre registry privée et relance les services via docker stack deploy, ainsi qu’un script rollback.sh capable de revenir à la version précédente en cas d’incident. Chacun de ces scripts est commenté en en-tête et documenté dans le README de déploiement.  Je définis ensuite deux environnements de tests :   * Intégration CI : un docker-compose.ci.yml instancie la base de tests et l’API, exécute la suite Jest/Supertest et génère un rapport de couverture de code. * Recette client : un cluster Kubernetes (Minikube en local ou environnement cloud sandbox) orchestré par Helm charts, dans lequel je déploie une version pré-production de BlockLumen. La procédure d’exécution des tests d’intégration (via Postman/Newman), des tests end-to-end (Cypress) et de la recette client (checklist de validation fonctionnelle et UX) est décrite pas à pas dans mon document « Déploiement & Tests ».   Enfin, j’instaure un système de veille technologique : j’abonne notre canal Slack aux flux RSS officiels de Docker, Kubernetes et TypeORM ; je supervise les alertes CVE liées à Node.js et MySQL via GitHub Dependabot ; et je planifie chaque mois une revue de sécurité pour intégrer les correctifs critiques (CORS, chiffrement TLS, rotation des clés). | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Je m’appuie sur :   * Scripts Bash & Helm charts pour automatiser le build, le push et le déploiement. * Docker Compose (fichiers docker-compose.prod.yml, docker-compose.ci.yml) et Docker Swarm ou Kubernetes (Minikube ou cluster cloud). * CI/CD avec GitLab CI (ou GitHub Actions) pour enchaîner build, tests, scans de vulnérabilité (Trivy) puis déploiement. * Postman/Newman et Cypress pour les tests d’intégration et de bout en bout. * Slack + RSS feeds + Dependabot pour la veille tech et sécurité. * Documentation centralisée dans un GitLab Wiki et un README en Markdown accompagné de diagrammes d’architecture déployés via Mermaid. | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| **Seul et en autonomie** | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | Cliquez ici pour taper du texte. | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/03/2025** | | **au** | **20/06/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  | | | | |
| **Activité-type** | **3** | | | **Développer une application sécurisée** | | | | |
| ***Exemple n°3***  | | | | ***Contribuer à la mise en production dans une démarche DevOps*** | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| 1. **Intégration des outils de qualité de code** J’installe et configure ESLint et Prettier dans le projet BlockLumen, avec des règles partagées via un fichier .eslintrc.js et un prettier.config.js. Dès qu’un développeur pousse du code, un hook Husky déclenche automatiquement la vérification et la mise en forme. En parallèle, j’ajoute un scanner SonarQube dans le pipeline pour mesurer la dette technique et la couverture de tests. 2. **Automatisation des tests** Je développe des suites de tests unitaires (Jest) et d’intégration (Supertest) pour chaque module back-end, ainsi que des tests end-to-end (Cypress) pour les parcours clés (authentification, trading, modules pédagogiques). Ces tests sont orchestrés automatiquement à chaque commit via notre pipeline CI. 3. **Écriture et maintien des scripts CI** Je crée un fichier gitlab-ci.yml (ou .github/workflows/ci.yml) décrivant trois jobs : lint, test, build-and-deploy. Chaque job s’exécute dans un conteneur Docker dédié, assure l’installation des dépendances, le lancement des linters, des tests, puis le packaging des artefacts si tout est vert. 4. **Configuration du serveur d’intégration continue** J’installe et paramètre un runner GitLab (ou GitHub Actions runner) sur notre VM de build : je définis les variables d’environnement sécurisées, les accès au registry Docker et à la base de données de test. Je configure également l’archivage des artefacts et des rapports de couverture pour chaque pipeline. 5. **Interprétation des rapports CI** À chaque exécution de pipeline, j’examine les tableaux de bord SonarQube pour identifier les nouvelles vulnérabilités et les règles violées. Je consulte le rapport Jest pour vérifier la couverture (target ≥ 80 %), et je partage un résumé quotidien dans notre canal Slack DevOps. 6. **Documentation technique des outils** Je rédige un guide en français (niveau B1 CECRL pour l’anglais) expliquant l’installation et l’usage d’ESLint, Jest, SonarQube et GitLab CI. Ce document inclut des captures d’écran des interfaces, des exemples de configuration et des liens vers la documentation officielle. | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| * Outils de qualité de code : ESLint, Prettier, Husky, SonarQube Community Edition * Automatisation de tests : Jest, Supertest, Cypress * CI/CD : GitLab CI (ou GitHub Actions), runner Docker, registry Docker privé * Infrastructure : conteneurs Docker orchestrés par GitLab Runner, VM de build Ubuntu 22.04 * Communication : Slack pour alertes pipeline, Confluence/GitLab Wiki pour la documentation | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| **Seul en autonomie** | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association**  | | | | | | ***Idem Creative Arts School*** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | |  | Cliquez ici pour taper du texte. | | | | | |
| **Période d’exercice** | |  | **Du** | | **01/03/2025** | | **au** | **20/06/2025** |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

|  |  |  |
| --- | --- | --- |
| **Titres, diplômes, CQP, attestations de formation** | | |
|  | | |
| *(facultatif)* | | |
| **Intitulé** | **Autorité ou organisme** | **Date** |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |

|  |
| --- |
| **Déclaration sur l’honneur** |
|  |
|  |

*LOPEZ Nicolas*

Je soussigné(e) [prénom et nom] ,

déclare sur l’honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l’auteur(e) des réalisations jointes.

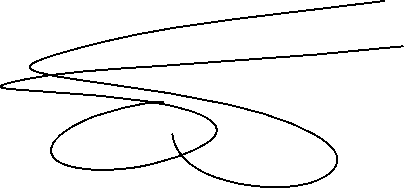
*Perpignan*

16/06/2025

Fait à le

pour faire valoir ce que de droit.

Signature :



|  |
| --- |
| **Documents illustrant la pratique professionnelle** |
|  |
| *(facultatif)* |
| **Intitulé** |
| Annexe 01 : Cahier des charges simplifiée – Page 32 |
| Annexe 01.1 : Concepts évoqués et dictionnaire de donnée – Page 33 |
| Annexe 02 : Zoning, Wireframe et Maquette – Page 33 |
| Annexe 03 : Responsive design & adaptation multi-supports – Page 34 |
| Annexe 04 : config tailwind – Page 35 |
| Annexe 05 : POO et SRP ( responsabilité unique ) – Page 36 |
| Annexe 06 : Sécurisation des composants serveur – Page 36 |
| Annexe 07 : Documentation du code source – Page 37 |
| Annexe 08 : Persona, User Journey Map, Zoning, Wireframe, Maquette Prototype, étude concurrence – Page 38 |
| Annexe 09 : Persona – Page 38 |
| Annexe 10 : User Journey Map – Page 39 |
| Annexe 11 : MCD BlockLumen – Page 40 |
| Annexe 12 : MPD BlockLumen – Page 41 |
| Annexe 13 : Fixtures BlockLumen – Page 42 |
| Annexe 14 : Controller avec TypeORM > UserController.ts – Page 43 |
| Annexe 15 : Routes > User.ts – Page 44 |

|  |
| --- |
| **Annexes** |
|  |
| *(Si le RC le prévoit)* |
| **Annexe 01 : Cahier des charges simplifiée** |

**Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.Annexe 01.1 : Concepts évoqués et dictionnaire de données**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.Annexe 02 : Zoning, Wireframe et Maquette**

**Annexe 03 : Responsive design & adaptation multi-supports**

Container fluide et centrage :

<div className="flex flex-col items-center container mx-auto p-4 font-aeonik">

Grilles adaptatives via breakpoints Tailwind :

<div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-4">

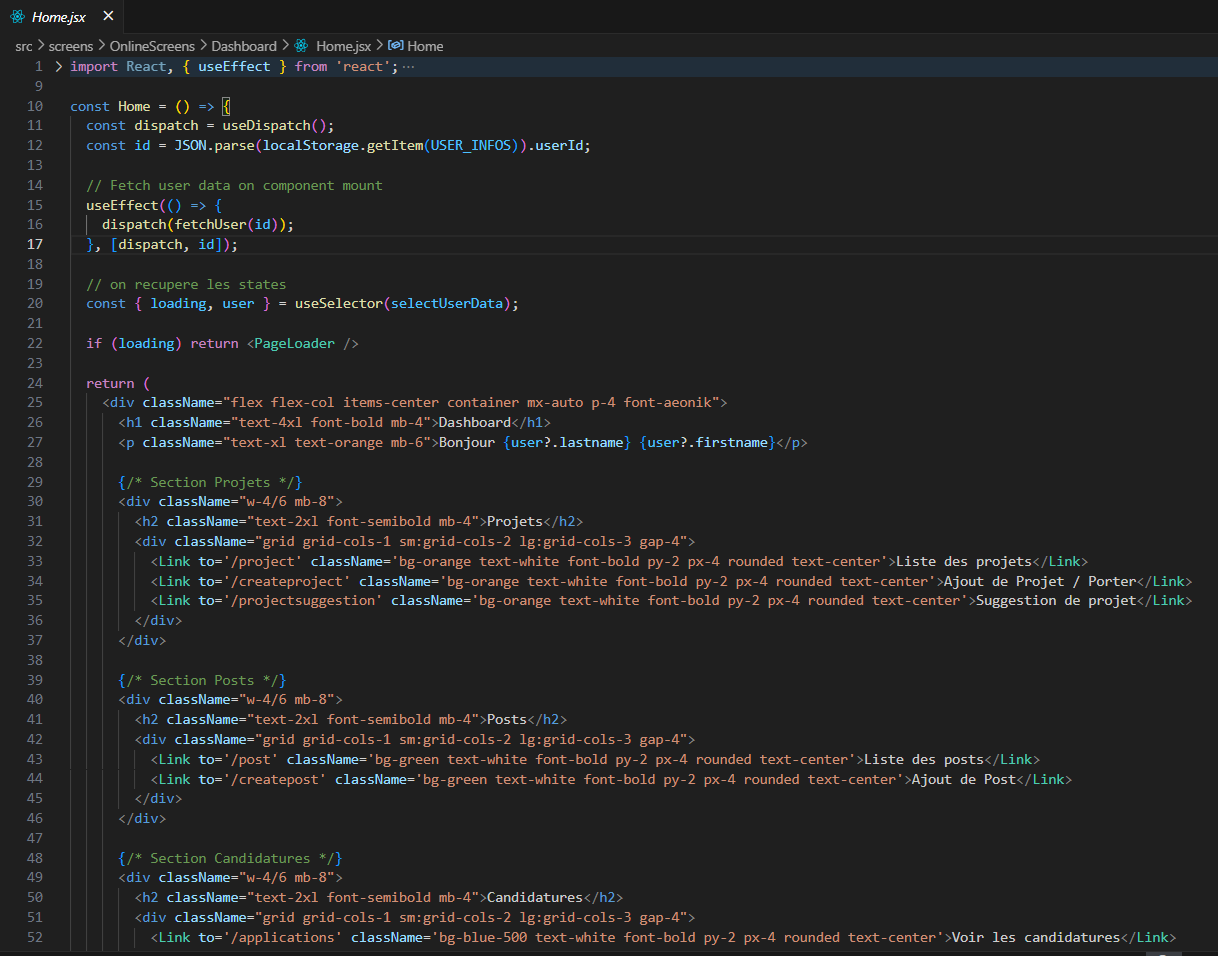
Largeur des blocs modulable :

<div className="w-4/6 mb-8">

Typographie et espacement cohérents :

La classe font-aeonik applique la police définie dans la charte Tailwind

Les marges (mb-4, mb-6, mb-8) et tailles de texte (text-4xl, text-2xl, text-xl) sont proportionnées pour conserver lisibilité et hiérarchie visuelle sur tous les supports.



**Annexe 04 : config tailwind**

Une image contenant texte, capture d’écran, affichage, logiciel

Le contenu généré par l’IA peut être incorrect.

**Annexe 05 : POO et SRP ( responsabilité unique )**

Ce bloc montre comment la logique métier est découpée en objets et méthodes à responsabilité unique (SRP).

Une image contenant texte, capture d’écran, logiciel, Police

Le contenu généré par l’IA peut être incorrect.

**Annexe 06 : Sécurisation des composants serveur**

L’extrait de CalculerPRTDocVente ou l’on vérifie l’existence de la commande et lève une erreur .

Ce code illustre la validation d’entrées et la gestion d’erreur avant tout traitement.

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.

**Annexe 07 : Documentation du code source**

Le **bloc de commentaires** en tête du codeunit (lignes 1–30) qui recense les révisions NSC et MIGRATION, ainsi que le commentaire juste avant procedure CalculerPR .

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.

**Annexe 08 : Persona, User Journey Map, Zoning, Wireframe, Maquette Prototype, étude concurrence**  
Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

**Annexe 09 : Persona**

Une image contenant texte, ordinateur, Visage humain, capture d’écran

Le contenu généré par l’IA peut être incorrect.

**Annexe 10 : User Journey Map**

Une image contenant texte, capture d’écran, logiciel, nombre

Le contenu généré par l’IA peut être incorrect.

**Une image contenant texte, diagramme, Plan, capture d’écran

Le contenu généré par l’IA peut être incorrect.Annexe 11 : MCD BlockLumen**

Annexe 12 : MPD BlockLumen

Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

**Annexe 12 : MPD BlockLumen**

Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

**Annexe 13 : Fixtures BlockLumen**

Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

**Annexe 14 : Controller avec TypeORM > UserController.ts**

Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Le contenu généré par l’IA peut être incorrect.

**Annexe 15 : Routes > user.ts**

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.