BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2026**

ANNEXE VII-1-B : Fiche descriptive de réalisation professionnelle (recto)

Épreuve E6 - Conception et développement d'applications (option SLAM)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE

Nom, prénom : LE ROUZIC Erwann	**N° candidat :** 2545861235
Épreuve ponctuelle ☒ **Contrôle en cours de formation** ☐	**Date:** 18/04/2026

Organisation support de la réalisation professionnelle
Campus Ermitage

Intitulé de la réalisation professionnelle
Application de rencontre de joueurs de jeu vidéo - GameConnect

Période de réalisation : 19 Janvier 2025 à 01 Septembre 2025 **Lieu :** Campus ermitage 47, Agen

Modalité : ☐ **Seul(e)** ☒ **En équipe**

Compétences travaillées

☒ Concevoir et développer une solution applicative

☐ Assurer la maintenance corrective ou évolutive d'une solution applicative

☒ Gérer les données

Conditions de réalisation[1] **(ressources fournies, résultats attendus)**

Ressources fournies :
— Cahier des charges pédagogique fourni par le commanditaire fictif DevDuJour décrivant la plateforme sociale d'e-sport attendue.
— Code source initial de la plateforme (squelette React et API FastAPI) servant de base au développement en équipe.
— Dépôt GitHub partagé pour la gestion de version et le travail collaboratif à trois.
— Environnement de développement local (Node.js, Python 3, MySQL) et outils de gestion de projet Agile (Trello, Scrum).

Résultats attendus :
— Application web fonctionnelle de mise en relation de joueurs, déployée et utilisable depuis un navigateur.
— Code source versé sur GitHub (https://github.com/R1Sobriquet/Esportapp), documenté et organisé en modules frontend et API.
— Base de données MySQL normalisée stockant les profils utilisateurs, les jeux, les messages et les matchs.
— API REST sécurisée (FastAPI, JWT) exposant les opérations de gestion des utilisateurs, du catalogue de jeux, du matching et de la messagerie.
— Tests unitaires et fonctionnels démontrant la fiabilité des modules développés et la conformité aux exigences de sécurité (OWASP).
— Documentation technique et utilisateur complète (README, schéma de base de données, cahier des spécifications).

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.

Description des ressources documentaires, matérielles et logicielles utilisées[2]

Ressources documentaires :
— Documentation officielle React 18, Vite et Tailwind CSS.
— Documentation officielle FastAPI, SQLAlchemy, PyJWT et bcrypt pour la partie backend.
— Documentation Swagger / OpenAPI générée automatiquement par FastAPI pour la description de l'API.
— Bonnes pratiques OWASP Top 10 pour la sécurisation de l'application web.
— Fichiers README.md du projet (API et frontend), cahier des spécifications et schéma de base de données, rédigés au fil des itérations.

Ressources matérielles :
— Postes de développement Windows 11 pour les trois membres de l'équipe.
— Serveur Windows hébergeant la base de données MySQL partagée au sein du campus.
— Réseau local du Campus Ermitage pour les tests d'intégration et les démonstrations.

Ressources logicielles :
— IDE : Visual Studio Code.
— Langages : JavaScript / JSX (frontend), Python 3 (backend), SQL (base de données).
— Frontend : React 18.2, Vite, Tailwind CSS.
— Backend : FastAPI, Uvicorn, SQLAlchemy, Pydantic, PyJWT, bcrypt.
— Base de données : MySQL 8.0 (10+ tables normalisées), hébergée sur serveur Windows.
— Gestion de version et collaboration : Git, hébergement GitHub.
— Gestion de projet Agile / Scrum : Trello pour le suivi du backlog et des sprints.
— Tests et validation : pytest pour l'API, Postman pour le test manuel des endpoints, outils OWASP pour la vérification de la sécurité.

Modalités d'accès aux productions[3] **et à leur documentation**[4]
— Dépôt du code source (frontend React et API FastAPI) : https://github.com/R1Sobriquet/Esportapp/tree/main
— Documentation d'installation et d'utilisation : fichiers README.md à la racine du dépôt (frontend) et du dossier API.
— Drive pour la documentation : https://drive.google.com/drive/folders/1ILVmWCIxEr1FgiHuAehVqiKJrqGdpSaS (contient : schéma de base de données, cahier des spécifications, fiche E6, captures d'écran et tableau Trello du projet).
— Documentation de l'API générée automatiquement (Swagger UI) accessible sur l'endpoint /docs de FastAPI lors de l'exécution locale.
— Application déployable localement à partir du dépôt GitHub : instructions d'installation et d'exécution détaillées dans les README.

2 Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

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.

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.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2026**

ANNEXE VII-1-B : Fiche descriptive de réalisation professionnelle (verso, éventuellement pages suivantes)

Épreuve E6 - Conception et développement d'applications (option SLAM)

Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs

Contexte et besoin

La réalisation a été menée dans le cadre de ma formation BTS SIO option SLAM au Campus Ermitage, en équipe de trois étudiants, dont j'ai assumé les rôles de chef de projet et de Scrum Master. Le projet répond à la demande d'un commanditaire fictif, l'entreprise DevDuJour, spécialisée dans l'organisation d'événements e-sport et le coaching de joueurs. Le besoin exprimé est de faciliter la mise en relation de joueurs compatibles : les gamers perdent aujourd'hui un temps considérable à chercher des coéquipiers partageant les mêmes jeux, niveaux de compétence et disponibilités sur des forums et réseaux sociaux dispersés.

L'objectif était de concevoir une plateforme web centralisée, GameConnect, permettant d'automatiser la recherche de coéquipiers via un algorithme de compatibilité, de faciliter la communication entre joueurs matchés et de sécuriser les échanges et les données personnelles.

Méthodologie de travail

Le projet a été conduit en méthode Agile Scrum, avec des sprints courts et des réunions régulières (daily stand-up, revues de sprint, rétrospectives). En tant que Scrum Master, j'ai animé les cérémonies, géré le backlog produit sur Trello et veillé à la répartition équilibrée des tâches entre les membres de l'équipe. Le code a été versé sur un dépôt GitHub partagé, avec une stratégie de branches par fonctionnalité et des pull requests croisées pour la revue de code.

Une séparation claire frontend / backend a été posée dès la conception, afin de permettre un développement en parallèle et de faciliter la maintenance et les évolutions futures. Chaque fonctionnalité a été développée de manière incrémentale : définition dans le backlog, implémentation côté API, implémentation côté interface, tests, puis intégration.

Objectifs et fonctionnalités

GameConnect a pour vocation d'offrir aux joueurs une solution unique pour gérer leur identité gaming, trouver des coéquipiers adaptés et échanger en toute sécurité. Les principaux objectifs fonctionnels sont :

— Mettre en relation des joueurs compatibles en fonction de leurs jeux, niveaux de compétence et disponibilités.

— Créer une communauté active autour des différents jeux via des espaces de discussion dédiés.

— Permettre aux utilisateurs de gérer leur identité gaming (profils, statistiques, préférences, intégrations Discord / Steam / Twitch).

— Automatiser la recherche de coéquipiers grâce à un algorithme de compatibilité produisant un score de 0 à 100%.

— Sécuriser les données personnelles et les échanges entre utilisateurs (authentification JWT, hachage des mots de passe, requêtes préparées).

— Faciliter la communication entre joueurs matchés via une messagerie intégrée.

Architecture fonctionnelle

L'application repose sur cinq grands modules fonctionnels, chacun exposé par l'API et consommé par l'interface React :

1. Gestion des profils : inscription, connexion, authentification par JWT (expiration 7 jours), profils gaming détaillés et intégrations de comptes sociaux (Discord, Steam, Twitch).
2. Catalogue de jeux : bibliothèque personnalisée par utilisateur, avec niveaux de compétence, rangs et statistiques. Ajout et suppression de jeux directement depuis le profil.
3. Matching intelligent : algorithme de compatibilité basé sur les jeux communs, les niveaux, les disponibilités et les préférences, restituant un score de 0 à 100 %.
4. Messagerie sécurisée : chat direct entre utilisateurs matchés, avec historique des conversations, notifications et suppression logique (soft delete) des messages.

5. Frontend moderne : interface réactive et responsive (Pages Games, Home, Legal, Login, Matching, Messages, Privacy, Profile, Register, Terms, composants Avatar, Footer, Toast, WelcomeMessage).

Architecture technique

GameConnect est construit sur une architecture trois tiers avec séparation stricte entre la présentation, la logique métier et les données :

— Couche présentation : application React 18.2 construite avec Vite et stylisée avec Tailwind CSS. Les appels à l'API sont centralisés dans un module de services.

— Couche métier : API REST développée en Python avec FastAPI, documentée automatiquement via Swagger / OpenAPI. Authentification par JWT, hachage des mots de passe avec bcrypt.

— Couche de données : base MySQL 8.0 comportant plus de 10 tables normalisées (utilisateurs, jeux, associations joueur-jeu, messages, matchs, etc.) hébergée sur un serveur Windows.

Les échanges entre le frontend et l'API se font via des requêtes HTTP JSON. Toutes les requêtes à la base de données sont paramétrées (requêtes préparées), ce qui prévient les injections SQL conformément aux préconisations OWASP.

Fonctionnalités sur lesquelles j'ai travaillé

1. Développement du système de messagerie sécurisée (API, interface, soft delete des messages, historique).
2. Frontend moderne : implémentation des pages Games, Home, Legal, Login, Matching, Messages, Privacy, Profile, Register, Terms et des composants Avatar, Footer, Toast, WelcomeMessage.
3. Catalogue de jeux : gestion de l'ajout et de la suppression de jeux directement depuis le profil utilisateur.
4. Gestion des profils : création, modification et suppression des comptes et des informations associées.
5. Tests unitaires : écriture et extension de la suite de tests pour fiabiliser les modules développés.

Exigences techniques et qualité

— Performance : temps de réponse de l'API ciblé à moins de 100 ms, requêtes SQL optimisées, score Lighthouse supérieur à 95 sur le frontend.

— Sécurité : authentification par JWT avec expiration à 7 jours, hachage bcrypt des mots de passe, requêtes préparées pour la protection contre les injections SQL, application des bonnes pratiques OWASP Top 10.

— Qualité : tests automatisés (unitaires et fonctionnels), gestion d'erreurs robuste côté API, validation des entrées via Pydantic, documentation technique à jour.

— Compatibilité : interface compatible web et mobile grâce à un design responsive.

Déploiement

L'application est conçue pour être déployée sur une infrastructure séparée en trois tiers : le frontend React compilé est servi par un serveur web, l'API FastAPI tourne derrière un serveur ASGI (Uvicorn) et la base MySQL est hébergée sur un serveur Windows dédié. Un déploiement cible sur une machine virtuelle Debian 12 avec Nginx en reverse proxy est prévu pour l'environnement de production pédagogique, permettant de valider la chaîne complète en conditions réelles.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2026**

ANNEXE VII-1-B : Fiche descriptive de réalisation professionnelle (recto)

Épreuve E6 - Conception et développement d'applications (option SLAM)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE

Nom, prénom : LE ROUZIC Erwann	**N° candidat :** 2545861235
Épreuve ponctuelle ☒ **Contrôle en cours de formation** ☐	**Date :** 08 juin 2026

Organisation support de la réalisation professionnelle
Entreprise MyDesyn

Intitulé de la réalisation professionnelle
EnrichFlow – Application desktop de prévision des commandes par pipeline ETL et modèles statistiques

Période de réalisation : 02 Septembre 2025 à 23 Novembre 2025 **Lieu :** Marmande

Modalité : ☒ **Seul(e)** ☐ **En équipe**

Compétences travaillées

☒ Concevoir et développer une solution applicative

☒ Assurer la maintenance corrective ou évolutive d'une solution applicative

☒ Gérer les données

Conditions de réalisation[1] **(ressources fournies, résultats attendus)**

Ressources fournies :
— Jeu de données de commandes 2024 (extrait de base SQL Server, disponible également en format CSV)
— Cahier des charges pédagogique BTS SIO SLAM portant sur la prévision des commandes journalières
— Accès à une base SQL Server de test
— Environnement de développement Python et dépôt GitHub pour la gestion du projet

Résultats attendus :
— Développement d'une application autonome de prévision des commandes, utilisable par un non-développeur
— Chargement et exploitation des données depuis un fichier CSV ou une base SQL Server
— Traitement automatisé des données (nettoyage, agrégation, enrichissement)
— Mise en œuvre et comparaison de plusieurs modèles de prévision
— Interface graphique simple et moderne, avec possibilité de déploiement sous forme d'exécutable

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.

Description des ressources documentaires, matérielles et logicielles utilisées[2]

Ressources documentaires :
— Documentation officielle Python 3.13, pandas, numpy, scikit-learn, matplotlib, seaborn.
— Documentation CustomTkinter (composants, thèmes, intégration avec Tkinter et le threading).
— Documentation Microsoft ODBC Driver 17 for SQL Server et SQLAlchemy.
— Documentation PyInstaller (mode onefile, hidden imports, fichiers .spec).
— README.md du projet, rédigé au fil des itérations, et fichier .env.example servant de modèle de configuration.

Ressources matérielles :
— Poste de développement Windows 11
— Serveur Microsoft SQL Server d'entreprise

Ressources logicielles :
— Langage : Python 3.13.
— Bibliothèques : pandas ≥ 2.1, numpy ≥ 1.24, scikit-learn ≥ 1.3, matplotlib ≥ 3.7, seaborn ≥ 0.12, scipy, statsmodels, customtkinter ≥ 5.2, pyodbc ≥ 5.0, sqlalchemy ≥ 2.0, python-dotenv ≥ 1.0, pydantic ≥ 2.0.
— Packaging : PyInstaller ≥ 6.0.
— IDE : PyCharm Professional.
— Gestion de version : Git, hébergement GitHub.
— Base de données : Microsoft SQL Server avec pilote ODBC Driver 17 for SQL Server.

Modalités d'accès aux productions[3] **et à leur documentation**[4]

— Dépôt du code source : https://github.com/R1Sobriquet/pythonPipelineEnrichissement
— Documentation d'installation et d'utilisation : fichier README.md à la racine du dépôt.
— Fichier de configuration modèle : .env.example (à copier en .env et renseigner DATA_SOURCE et identifiants SQL Server).
— Drive pour la documentation : https://drive.google.com/drive/folders/1zjqrWsb4drumdoYGKJ-WwVI2k6zO9DK9?usp=sharing (Contient : Schéma de Base de données, Cahier des spécifications, Feuille de route, Cahier de tests, Guide d'utilisation)
— Exécutable packagé EnrichFlow.exe disponible en démonstration lors de la soutenance.
— Identifiants et informations de connexion SQL Server présent sur le drive

2 Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

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.

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.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2026**

ANNEXE VII-1-B : Fiche descriptive de réalisation professionnelle (verso, éventuellement pages suivantes)

Épreuve E6 - Conception et développement d'applications (option SLAM)

Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs

Contexte et besoin
La réalisation a été développée dans le cadre de mon alternance au sein de l'entreprise MyDesyn. Il s'agit d'un prototype applicatif destiné à explorer la faisabilité d'un projet client : un système de prévision des commandes de recharge en billets pour un parc de distributeurs automatiques de billets (DAB). Le client, exploitant d'un réseau d'ATM, souhaite optimiser le réapprovisionnement de ses machines en anticipant les volumes à livrer par DAB et par jour, afin de réduire les ruptures (machine vide) comme les immobilisations de trésorerie (machine surchargée) et de rationaliser les tournées de collecte. Le prototype EnrichFlow devait démontrer, sur un jeu de données représentatif, la faisabilité technique d'un pipeline complet : chargement des données, nettoyage, enrichissement, entraînement de plusieurs modèles de prévision et évaluation comparative, le tout packagé en application desktop autonome pour être présentée et manipulable par les décideurs du client sans compétence technique.

Méthodologie de travail
La démarche suivie a été itérative et incrémentale. Chaque étape du pipeline a d'abord été développée et validée en isolation (interface en ligne de commande), avant d'être intégrée à l'interface graphique puis au packaging final. Le code a été versionné sur GitHub tout au long du projet, avec des commits atomiques documentant chaque évolution. Une séparation claire des responsabilités (moteur de traitement et interface dans deux packages distincts) a été posée dès la conception, afin de faciliter la maintenance, les tests et les évolutions futures de la solution.

Objectifs et fonctionnalités
L'objectif de l'application EnrichFlow est de fournir une solution autonome permettant d'analyser des données de commandes et de réaliser des prévisions fiables à partir de différentes sources de données.
La solution a pour vocation d'automatiser l'ensemble du processus de traitement, depuis le chargement des données jusqu'à l'évaluation des modèles de prévision.
L'application propose une interface graphique simple, permettant à un utilisateur non développeur de piloter les traitements, d'analyser les résultats et de comparer les prévisions sans intervention technique.

Architecture fonctionnelle
L'application repose sur un pipeline ETL structuré en quatre étapes successives, chacune pouvant être exécutée indépendamment depuis l'interface :

1. Ingestion et nettoyage des données
Chargement des données depuis un fichier CSV ou une base SQL Server, normalisation des informations, suppression des données incohérentes et agrégation par date et article afin d'obtenir un jeu de données exploitable.

2. Enrichissement des données
Génération automatique de variables explicatives (informations temporelles, retards, moyennes mobiles) destinées à améliorer la qualité des analyses et des prévisions.

3. Analyse des données
Production de visualisations permettant d'identifier les tendances, les saisonnalités et les comportements de consommation.

4. Prévision et comparaison des modèles
Mise en œuvre de plusieurs modèles de prévision de référence, leur évaluation et la comparaison automatique des performances afin d'identifier le modèle le plus pertinent.

Architecture technique
La solution est développée en Python et repose sur une séparation claire entre la logique métier et l'interface graphique.
Le moteur de traitement (ETL, analyses, modèles) est regroupé dans un module dédié, tandis que l'interface graphique permet de piloter l'exécution du pipeline et de visualiser les résultats sans bloquer l'application.

L’architecture supporte deux sources de données interchangeables (CSV ou SQL Server), ce qui facilite à la fois le développement, les tests et le déploiement en environnement réel.

Déploiement
L’application est fournie sous la forme d’un exécutable Windows autonome, permettant son installation et son utilisation sur un poste client sans nécessiter d’environnement de développement ni l’installation de Python. Elle communique avec une base de données distante existante, utilisée pour l’accès et l’exploitation des données, assurant ainsi l’intégration de la solution dans un environnement applicatif réel.