BTS SERVICES INFORMATIQUES AUX ORGANISATIONS SESSION 2025 Épreuve E6 - Conception et développement d'applications (option SLAM)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE

N° réalisation : 2

Nom, prénom : Pecqueur Florian

N° candidat :02443860533

Épreuve ponctuelle ☒ **Contrôle en cours de formation** ☐

Date : 22/05/2025

Organisation support de la réalisation professionnelle

Dans le cadre d'un projet personnel, j'ai développé une application Python utilisant Pygame pour créer un jeu de type "Motus" en duel, avec enregistrement des joueurs et de leurs scores dans une base de données MySQL.

Intitulé de la réalisation professionnelle

Motus Duel : Jeu multijoueur local en Python (POO)

Période de réalisation : 01/04/2025 **Lieu :** Campus Diderot éducation Lille

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)

L'application **Motus Duel** est un programme développé en Python utilisant le module Pygame pour créer un jeu de type "Motus" en local, avec une interface graphique. L'application permet à deux joueurs de s'affronter en tour par tour : chaque joueur choisit un mot que l'autre doit deviner. Les lettres sont affichées dans des cases, colorées en fonction de leur justesse vert pour une lettre bien placée, jaune pour une lettre présente mais mal placée, gris pour une lettre absente.

Chaque joueur entre son nom via un écran d'accueil, et les informations (nom, scores, nombre de victoires et de défaites) sont enregistrées dans une base de données MySQL.

Le score est calculé en fonction de la difficulté du mot (plus le mot est long, plus il rapporte de points). Le jeu utilise une architecture en Programmation Orientée Objet (POO) et gère l'affichage graphique, la logique du duel et la persistance des données.

Description des ressources documentaires, matérielles et logicielles utilisées

Description des ressources documentaires, matérielles et logicielles utilisées

- Langage de programmation : Python 3
- Outils de gestion de base de données : MySQL Workbench
- Serveur local : MAMP (pour héberger la base de données MySQL sur localhost)
- IDE utilisé : Visual Studio Code
- Matériel : MacBook Pro M1

Modalités d'accès aux productions[3] et à leur documentation[4]
Le projet sera mis à disposition sur un dépôt GitHub personnel accessible à l'adresse suivante : https://github.com/PecqueurFlorian
Le dépôt contiendra l'ensemble du code source, les fichiers de documentation (structure de la base de données, description des classes), un export SQL, ainsi que des captures d'écran de l'interface utilisateur.

[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.
[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 2024

ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle (verso, éventuellement pages suivantes)

Épreuve E5 - Conception et développement d'applications (option SLAM)

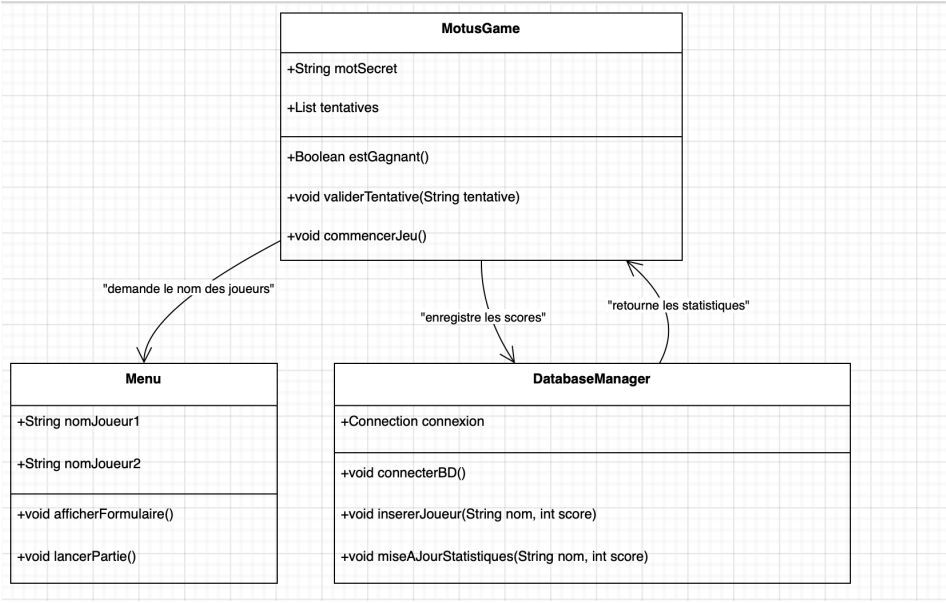
Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs

Ce projet Motus Duel consiste à développer une application Python permettant à deux joueurs de s'affronter en tour par tour dans un jeu de type "Motus". Le projet utilise le module Pygame pour proposer une interface graphique interactive et attractive, et est entièrement développé selon les principes de la programmation orientée objet (POO). Les joueurs doivent deviner un mot choisi par leur adversaire, avec un système de couleurs (vert, jaune, gris) indiquant la justesse des lettres proposées.

Les noms des joueurs, leurs scores et les statistiques de parties sont enregistrés dans une base de données MySQL, hébergée localement via MAMP, ce qui permet de suivre l'historique des parties.

1°/ Schéma UML du projet

Le diagramme ci-dessous présente la structure orientée objet de l'application Motus Duel il a était réalise avec draw.io.
Il met en évidence les classes principales utilisées pour gérer le jeu, les joueurs, les mots à deviner ainsi que les interactions avec la base de données.



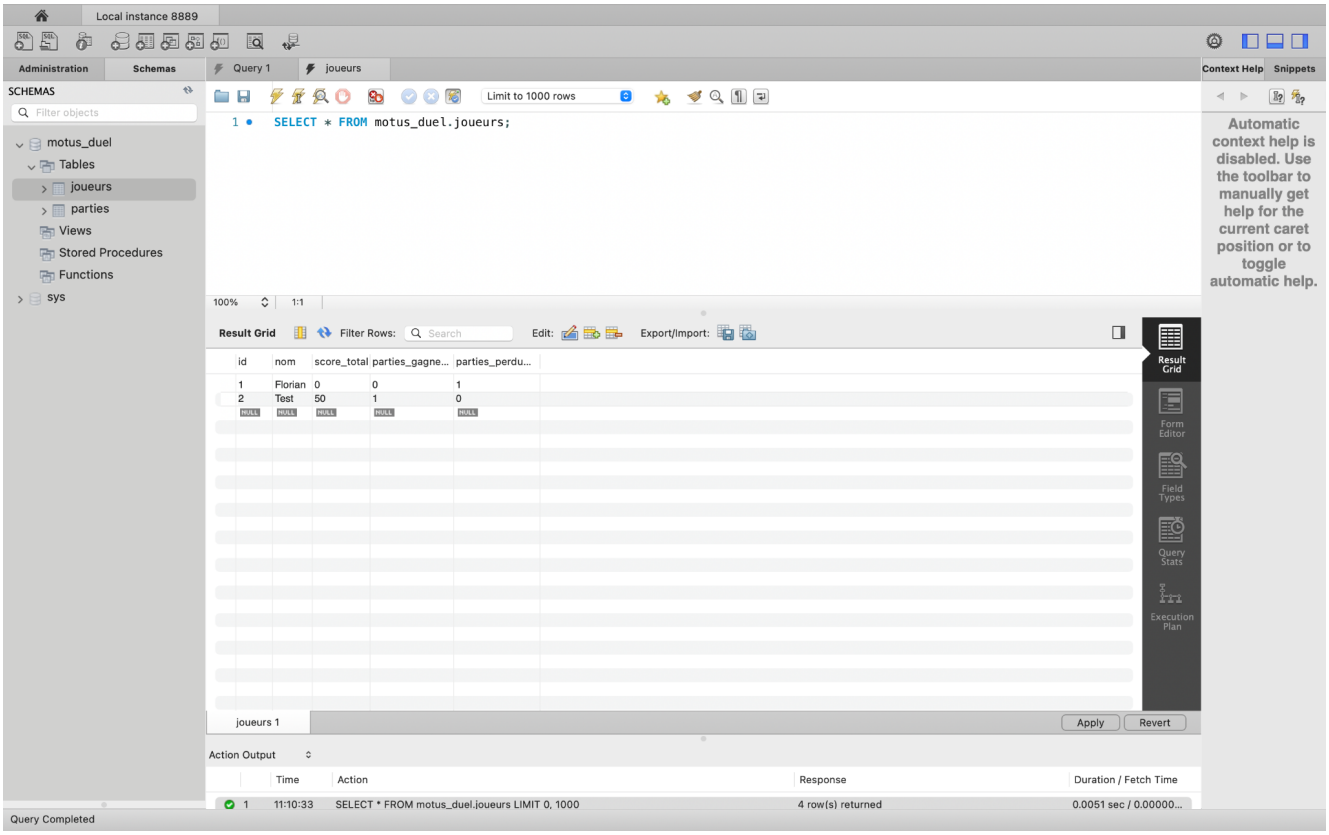
2°/ Création de la base de données avec les tables joueurs et parties.

Une base de données appelée motus_duel a été créée sur un serveur local MySQL via MAMP.
Cette base contient deux tables principales :

- Table joueurs : elle stocke les informations des participants, notamment leur identifiant (id_joueur), leur nom (nom_joueur), leur nombre de victoires et de défaites.
- Table parties : elle enregistre chaque duel effectué entre deux joueurs, le mot secret utilisé, le vainqueur, et le nombre de tentatives effectuées.

Un script SQL a été utilisé pour créer ces tables et assurer l'intégrité des données via des clefs primaires et des clefs étrangères (id_joueur lié à parties).
À terme, cette base permet de suivre les performances de chaque joueur et de conserver un historique des parties réalisées.

L'image ci-dessous montre un extrait de la table joueurs dans MySQL Workbench.
Elle permet de stocker les informations de chaque joueur, notamment leur nom, leur score total ainsi que le nombre de parties gagnées ou perdues.
Cette base est essentielle pour suivre la progression de chaque participant au sein du jeu Motus Duel.



3°/ Développement du module database.py pour gérer les interactions entre l'application et MySQL.

Le fichier database.py gère la connexion à MySQL et les échanges avec la base de données. Il permet d'insérer les joueurs, de mettre à jour leur score et de sauvegarder les résultats des parties. Toutes les opérations sont regroupées dans une classe DatabaseManager, respectant ainsi la programmation orientée objet.

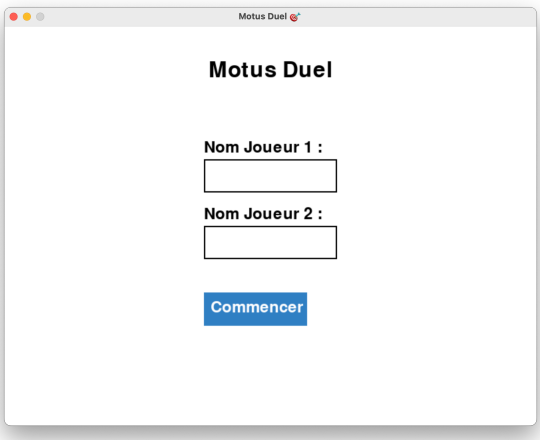
```python
import mysql.connector
class DatabaseManager:
    def __init__(self):
        self.connection = mysql.connector.connect(
            host="localhost",
            port=8889,
            user="root",
            password="root",
            database="motus_duel"
        )
        self.cursor = self.connection.cursor()
    def ajouter_joueur(self, nom):
        query = "INSERT INTO joueurs (nom) VALUES (%s)"
        self.cursor.execute(query, (nom,))
        self.connection.commit()
    def maj_score(self, nom, score, gagne):
        if gagne:
            query = """
                UPDATE joueurs
                SET score_total = score_total + %s, parties_gagnees = parties_gagnees + 1
                WHERE nom = %s
            """
        else:
            query = """
                UPDATE joueurs
                SET score_total = score_total + %s, parties_perdues = parties_perdues + 1
                WHERE nom = %s
            """
        self.cursor.execute(query, (score, nom))
        self.connection.commit()
    def fermer_connexion(self):
        self.cursor.close()
        self.connection.close()
```

4°/ Conception d'un écran d'accueil avec saisie du nom des deux joueurs et insertion en base

Un formulaire graphique a été conçu à l'aide de la bibliothèque Pygame pour permettre aux utilisateurs de saisir les noms des deux joueurs. Une fois validés, les noms sont automatiquement insérés dans la base de données via le module database.py.

```python
class Menu:
    def __init__(self):
        self.joueur1 = ""
        self.joueur2 = ""
        self.active_input1 = False
        self.active_input2 = False
        self.input_box1 = pygame.Rect(300, 200, 200, 50)
        self.input_box2 = pygame.Rect(300, 300, 200, 50)
        self.start_button = pygame.Rect(350, 400, 155, 50)
        self.db = DatabaseManager()
    def afficher(self):
        screen.fill(BLANC)
        titre = font.render("Motus Duel", True, NOIR)
        screen.blit(titre, (LARGEUR//2 - titre.get_width()//2, 50))
        txt_surface1 = small_font.render(self.joueur1, True, NOIR)
        pygame.draw.rect(screen, BLEU if self.active_input1 else NOIR, self.input_box1, 2)
        screen.blit(txt_surface1, (self.input_box1.x+5, self.input_box1.y+10))
        label1 = small_font.render("Nom Joueur 1 :", True, NOIR)
        screen.blit(label1, (self.input_box1.x, self.input_box1.y - 30))
        txt_surface2 = small_font.render(self.joueur2, True, NOIR)
        pygame.draw.rect(screen, BLEU if self.active_input2 else NOIR, self.input_box2, 2)
        screen.blit(txt_surface2, (self.input_box2.x+5, self.input_box2.y+10))
        label2 = small_font.render("Nom Joueur 2 :", True, NOIR)
        screen.blit(label2, (self.input_box2.x, self.input_box2.y - 30))
        pygame.draw.rect(screen, BLEU, self.start_button)
        start_text = small_font.render("Commencer", True, BLANC)
        screen.blit(start_text, (self.start_button.x + 10, self.start_button.y + 10))

```

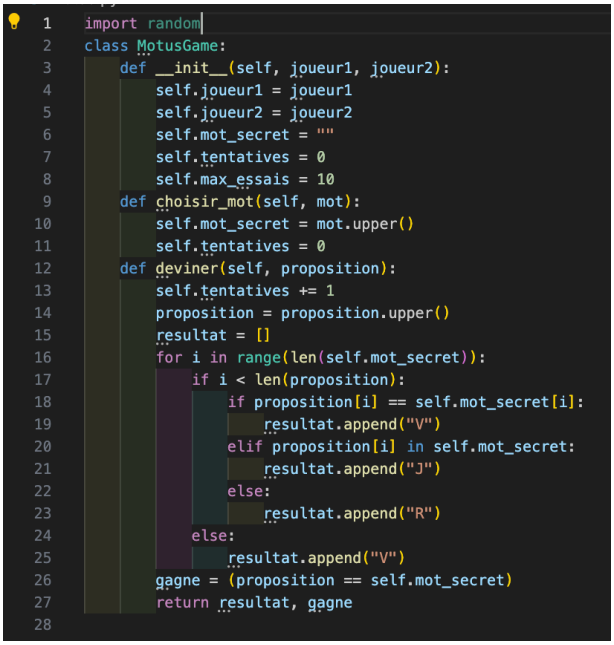


5°/Création de la logique du jeu

La classe MotusGame gère toute la logique du jeu. Chaque joueur choisit un mot à faire deviner à l'autre. Lors de son tour, un joueur propose un mot, et le programme vérifie pour chaque lettre si elle est :

- Correctement placée (couleur verte),
- Présente mais mal placée (couleur jaune),
- Absente (couleur rouge).

Le nombre de tentatives est limité. Après chaque proposition, les résultats sont affichés, et le tour passe à l'autre joueur si besoin. Cette gestion est entièrement réalisée en programmation orientée objet.



```python
import random
class MotusGame:
    def __init__(self, joueur1, joueur2):
        self.joueur1 = joueur1
        self.joueur2 = joueur2
        self.mot_secret = ""
        self.tentatives = 0
        self.max_essais = 10
    def choisir_mot(self, mot):
        self.mot_secret = mot.upper()
        self.tentatives = 0
    def deviner(self, proposition):
        self.tentatives += 1
        proposition = proposition.upper()
        resultat = []
        for i in range(len(self.mot_secret)):
            if i < len(proposition):
                if proposition[i] == self.mot_secret[i]:
                    resultat.append("V")
                elif proposition[i] in self.mot_secret:
                    resultat.append("J")
                else:
                    resultat.append("R")
            else:
                resultat.append("V")
        gagne = (proposition == self.mot_secret)
        return resultat, gagne

```

6°/Mise en place de l’interface graphique avec Pygame (titre, grille, affichage des lettres)

Une interface graphique plus complète est en cours de développement avec Pygame.
Cette interface doit permettre d’afficher un plateau de jeu visuel : un titre, une grille de lettres, ainsi que les couleurs pour les lettres correctement placées, mal placées ou absentes.
L’objectif est d’améliorer l’expérience utilisateur et de rendre la partie plus immersive.

7°/Connexion entre l’interface, la base de données, et les règles du jeu

Le projet prévoit également de connecter pleinement l’interface graphique, la base de données MySQL et la logique du jeu.
À chaque fin de partie, les résultats (gagnant, score, nombre de tentatives) seront enregistrés automatiquement en base, et les données pourront être affichées sur une interface dédiée aux scores des joueurs.