ENSAMc

Mustapha HAIN

Rapport du Projet Python Avancé

Département Informatique

ENSAMc

Application de Gestion des Notes des Étudiants

(Base de données SQLite et interface graphique Tkinter)

Ajout des notes

Cette fonctionnalité permet au professeur d’ajouter les notes des étudiants pour un module donné.

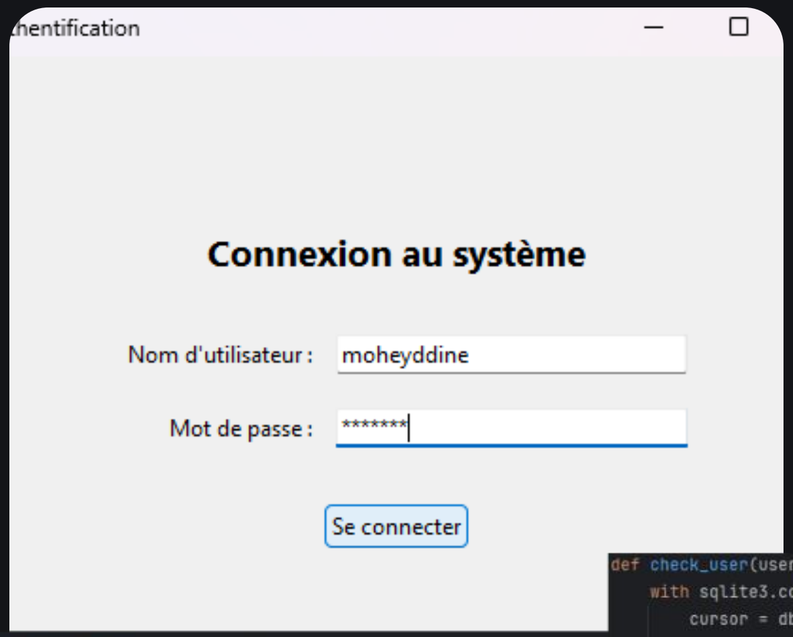
Modification d’une note

Cette option permet de mettre à jour la note d’un étudiant déjà enregistré.

Exportation des notes

Cette fonctionnalité permet d’exporter les notes des étudiants d’un module sélectionné vers un fichier CSV.

Page de connexion



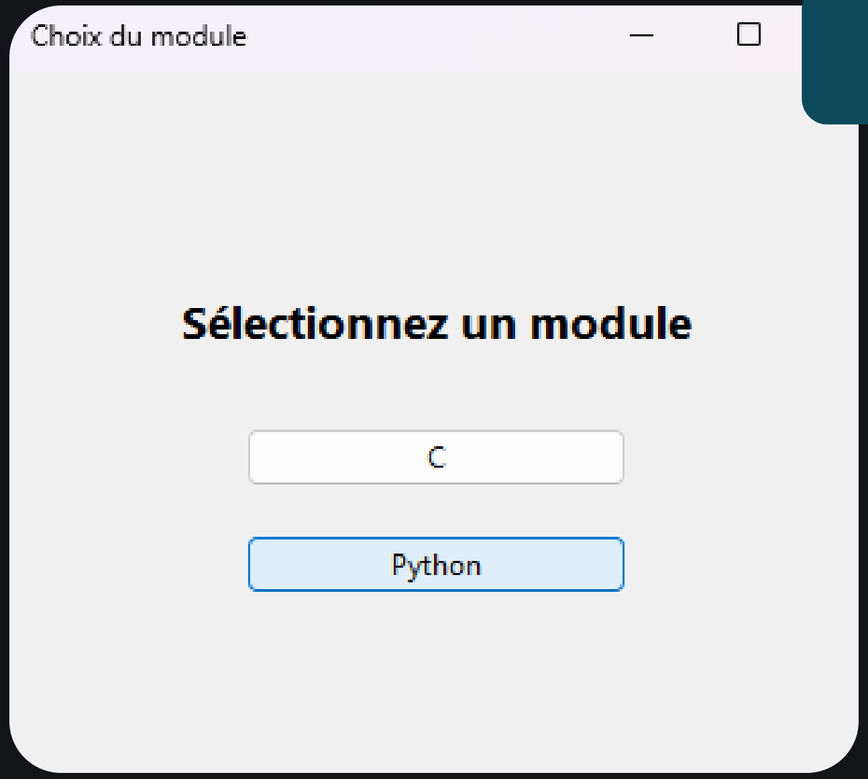
La page de connexion assure la sécurité de l'application.
Chaque professeur possède un nom d'utilisateur et un mot de passe enregistrés dans la base de données.
Après la saisie et la validation, le programme vérifie les informations dans la table professor.
En cas de réussite, l'utilisateur accède à la page de sélection du module ; sinon, un message d'erreur s'affiche.
Cette étape garantit que seuls les professeurs autorisés peuvent utiliser le système.

```python
def check_user(user, pswd):  1 usage
    with sqlite3.connect("base_de_donnees.db") as db:
        cursor = db.execute( sql: "SELECT * FROM professor WHERE username = ? AND password = ?",  parameters: (user, pswd))
        row = cursor.fetchone()

    if row:
        messagebox.showinfo( title: "Succès",  message: "Connexion réussie !")
        open_window_module()
    else:
        messagebox.showerror( title: "Erreur",  message: "Nom d'utilisateur ou mot de passe invalide !")
        open_window_login()
```

```python
def open_window_login():  2 usages
    window = Tk()
    window.geometry("500x350")
    window.title("Authentification")
    window.minsize( width: 400,  height: 300)

    frame = ttk.Frame(window, padding=40)
    frame.pack(expand=True)

    ttk.Label(frame, text="Connexion au système", font=("Segoe UI", 14, "bold")).grid(row=0, column=0, columnspan=2,
                                                                                      pady=20)
    ttk.Label(frame, text="Nom d'utilisateur :").grid(row=1, column=0, sticky="e", padx=10, pady=10)
    ttk.Label(frame, text="Mot de passe :").grid(row=2, column=0, sticky="e", padx=10, pady=10)

    username = StringVar()
    password = StringVar()

    ttk.Entry(frame, textvariable=username, width=30).grid(row=1, column=1)
    ttk.Entry(frame, textvariable=password, width=30, show="*").grid(row=2, column=1)

    ttk.Button(frame, text="Se connecter",
               command=lambda: [window.destroy(), check_user(username.get(), password.get())]
               ).grid(row=3, column=0, columnspan=2, pady=20)

    window.mainloop()
```

Cette fonction crée la fenêtre d'authentification de l'application.
Elle affiche deux champs de saisie pour le nom d'utilisateur et le mot de passe, ainsi qu'un bouton « Se connecter ».
Lorsqu'un utilisateur valide ses informations, la fonction appelle check_user() pour vérifier leur validité dans la base de données.
Elle constitue le point d'entrée principal du programme.

Page de sélection du module

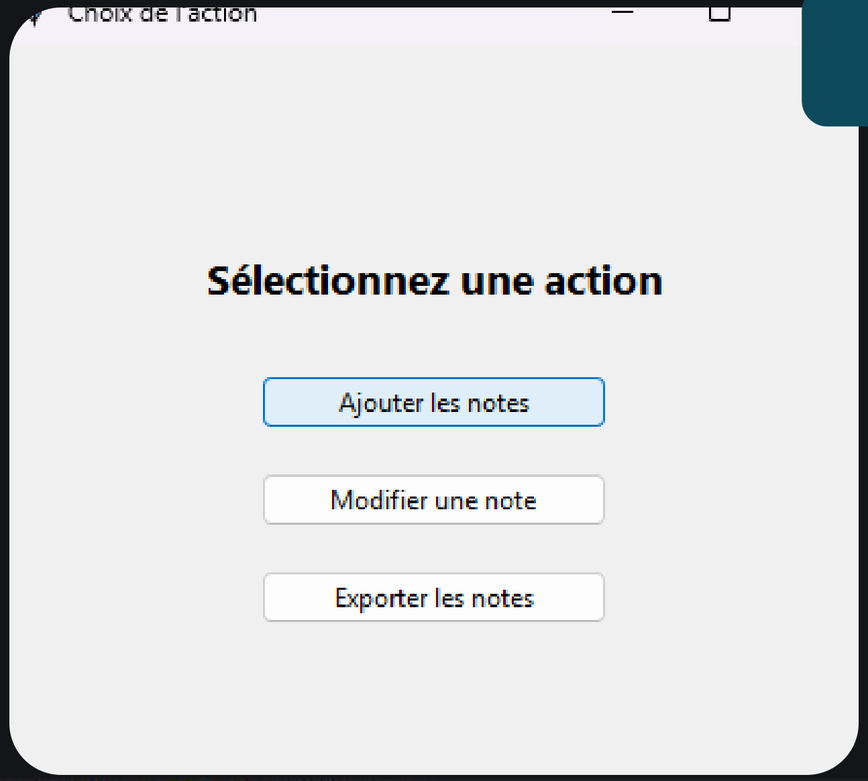


Après la connexion, l'utilisateur accède à la page de sélection du module.
Cette interface propose deux choix : C ou Python, correspondant aux matières enseignées.
Le professeur sélectionne le module dans lequel il souhaite gérer les notes.
Ce choix détermine la suite des actions possibles : ajout, modification ou exportation des notes des étudiants.

```python
def open_window_module():  5 usages
    window = Tk()
    window.title("Choix du module")
    window.geometry("400x300")
    window.minsize( width: 300,  height: 250)

    frame = ttk.Frame(window, padding=40)
    frame.pack(expand=True)

    ttk.Label(frame, text="Sélectionnez un module", font=("Segoe UI", 14, "bold")).pack(pady=20)
    ttk.Button(frame, text="C", command=lambda: [window.destroy(), open_window_action(1)], width=25).pack(pady=10)
    ttk.Button(frame, text="Python", command=lambda: [window.destroy(), open_window_action(2)], width=25).pack(pady=10)

    window.mainloop()
```

Cette fonction ouvre la fenêtre de sélection du module après la connexion du professeur.
Elle propose deux boutons correspondant aux modules C et Python.
Le choix du professeur détermine le module sur lequel il pourra ensuite ajouter, modifier ou exporter les notes.
Cette étape permet de diriger le professeur vers les actions spécifiques à chaque matière.

Page de sélection de l’action

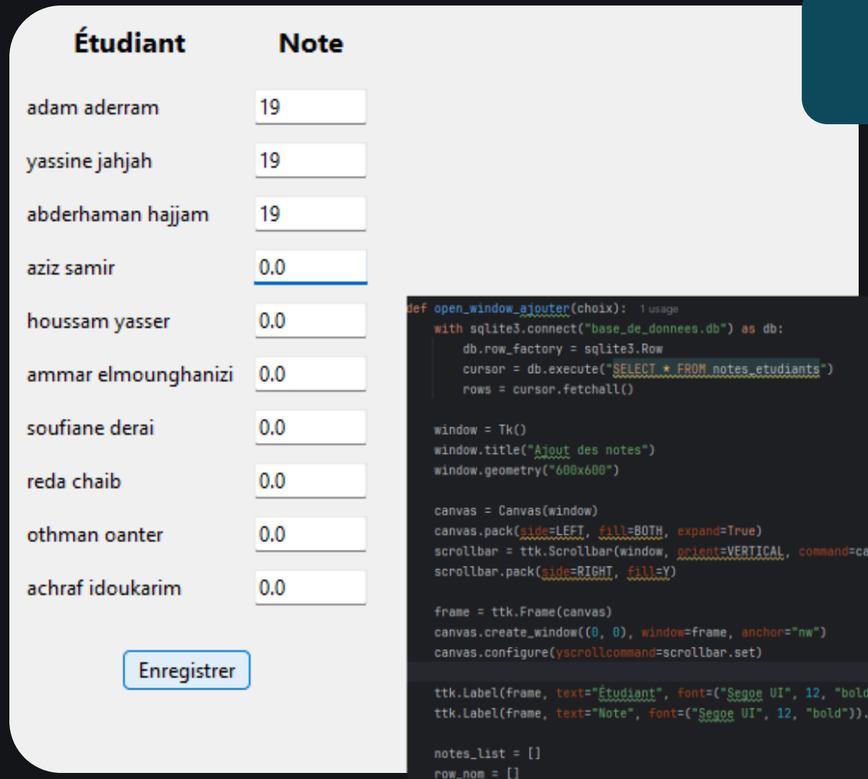


Une fois le module choisi, l’utilisateur est redirigé vers la page de sélection de l’action.
Cette interface propose trois options : ajouter les notes, modifier une note, ou exporter les notes.
Chaque bouton ouvre une nouvelle fenêtre correspondant à l’action choisie.
Cette étape permet au professeur de gérer facilement les notes des étudiants selon ses besoins.
Page de sélection de l’action

```python
def open_window_action(choix):  2 usages
    window = Tk()
    window.title("Choix de l'action")
    window.geometry("400x350")
    window.minsize( width: 300,  height: 300)

    frame = ttk.Frame(window, padding=40)
    frame.pack(expand=True)

    ttk.Label(frame, text="Sélectionnez une action", font=("Segoe UI", 14, "bold")).pack(pady=20)
    ttk.Button(frame, text="Ajouter les notes", command=lambda: [window.destroy(), open_window_ajouter(choix)],
               width=25).pack(pady=10)
    ttk.Button(frame, text="Modifier une note", command=lambda: [window.destroy(), open_window_modifier(choix)],
               width=25).pack(pady=10)
    ttk.Button(frame, text="Exporter les notes", command=lambda: [window.destroy(), open_window_export(choix)],
               width=25).pack(pady=10)

    window.mainloop()
```

Cette fonction ouvre la fenêtre de sélection de l’action pour le professeur, après qu’il ait choisi un module.
Elle propose trois options : ajouter les notes, modifier une note, ou exporter les notes.
Le professeur clique sur l’action souhaitée, ce qui ouvre la fenêtre correspondante pour gérer les notes des étudiants dans le module sélectionné.
Cette interface facilite la gestion complète des notes par le professeur.

Page d’ajout des notes

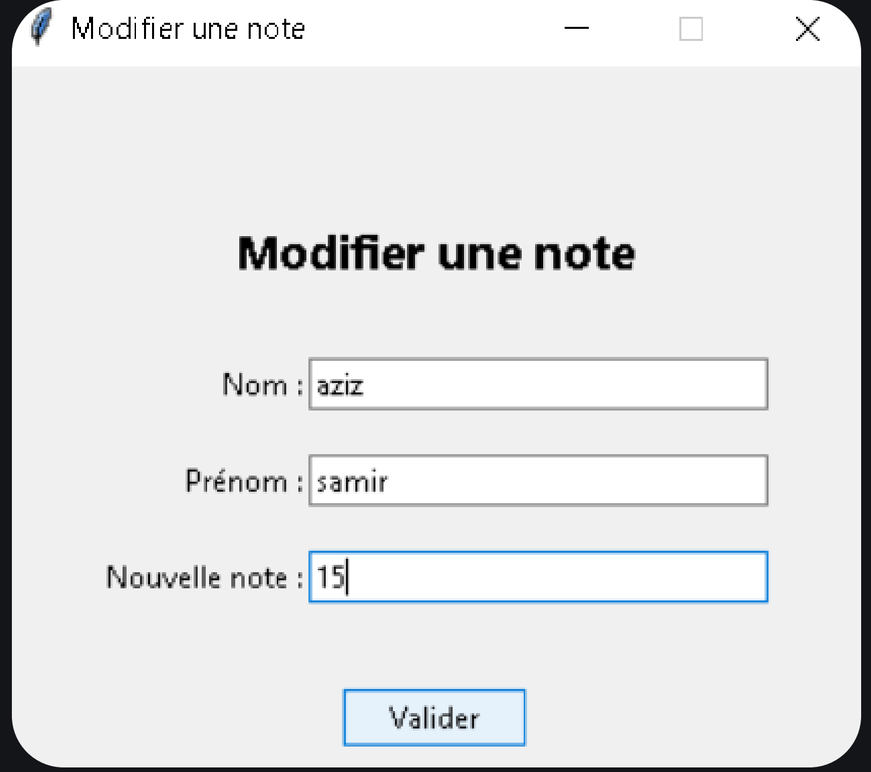


La page d’ajout des notes affiche la liste complète des étudiants inscrits dans la base de données.
Pour chaque étudiant, un champ de saisie permet d’entrer sa note dans le module sélectionné (C ou Python).
Une fois toutes les notes renseignées, le professeur clique sur « Enregistrer », et les données sont automatiquement sauvegardées dans la base SQLite.
Cette interface facilite une saisie rapide et organisée des résultats.

```python
def open_window_ajouter(choix):  1 usage
    with sqlite3.connect("base_de_donnees.db") as db:
        db.row_factory = sqlite3.Row
        cursor = db.execute("SELECT * FROM notes_etudiants")
        rows = cursor.fetchall()

    window = Tk()
    window.title("Ajout des notes")
    window.geometry("600x600")

    canvas = Canvas(window)
    canvas.pack(side=LEFT, fill=BOTH, expand=True)
    scrollbar = ttk.Scrollbar(window, orient=VERTICAL, command=canvas.yview)
    scrollbar.pack(side=RIGHT, fill=Y)

    frame = ttk.Frame(canvas)
    canvas.create_window((0, 0), window=frame, anchor="nw")
    canvas.configure(yscrollcommand=scrollbar.set)

    ttk.Label(frame, text="Étudiant", font=("Segoe UI", 12, "bold")).grid(row=0, column=0, padx=10, pady=10)
    ttk.Label(frame, text="Note", font=("Segoe UI", 12, "bold")).grid(row=0, column=1, padx=10, pady=10)

    notes_list = []
    row_nom = []

    for index, row in enumerate(rows):
        full_name = f"{row['nom']} {row['prenom']}"
        ttk.Label(frame, text=full_name).grid(row=index + 1, column=0, pady=5, padx=10, sticky="w")
        var = DoubleVar()
        ttk.Entry(frame, textvariable=var, width=10).grid(row=index + 1, column=1, pady=5)
        notes_list.append(var)
        row_nom.append((row["nom"], row["prenom"]))

    ttk.Button(frame, text="Enregistrer",
               command=lambda: [window.destroy(), add_grades(choix, row_nom, notes_list)]
               ).grid(row=len(row_nom) + 1, columnspan=2, pady=20)

    frame.update_idletasks()
    canvas.config(scrollregion=canvas.bbox("all"))

    window.mainloop()
```

Cette fonction ouvre la fenêtre d’ajout des notes pour le professeur, après la sélection d’un module.
Elle affiche la liste complète des étudiants avec un champ de saisie pour chaque note.
Le professeur peut entrer les notes pour tous les étudiants du module sélectionné, puis cliquer sur « Enregistrer ».
La fonction add_grades() sauvegarde automatiquement ces notes dans la base de données SQLite.
Cette interface facilite une saisie rapide et organisée des résultats par le professeur.

```python
def add_grades(choix, row_nom, notes_list):  1 usage
    with sqlite3.connect("base_de_donnees.db") as db:
        for index, (nom, prenom) in enumerate(row_nom):
            note = notes_list[index].get()
            if choix == 1:
                db.execute( sql: "UPDATE notes_etudiants SET C = ? WHERE nom = ? AND prenom = ?",  parameters: (note, nom, prenom))
            elif choix == 2:
                db.execute( sql: "UPDATE notes_etudiants SET python = ? WHERE nom = ? AND prenom = ?",  parameters: (note, nom, prenom))
        db.commit()

    messagebox.showinfo( title: "Succès",  message: "Les notes ont été enregistrées avec succès !")
    open_window_module()
```



Page de modification des notes

Cette fenêtre permet au professeur de saisir le nom, le prénom et la nouvelle note d'un étudiant.
Après avoir rempli les champs,
il clique sur « Valider » pour enregistrer la modification dans la base de données.

```python
def verify(nom, prenom, note, choix):  1 usage
    with sqlite3.connect("base_de_donnees.db") as db:
        result = db.execute( sql: "SELECT * FROM notes_etudiants WHERE nom = ? AND prenom = ?",  parameters: (nom, prenom)).fetchone()
        if result is None:
            messagebox.showerror( title: "Erreur",  message: "Étudiant introuvable !")
            open_window_module()
            return
        if choix == 1:
            db.execute( sql: "UPDATE notes_etudiants SET C = ? WHERE nom = ? AND prenom = ?",  parameters: (note, nom, prenom))
        elif choix == 2:
            db.execute( sql: "UPDATE notes_etudiants SET python = ? WHERE nom = ? AND prenom = ?",  parameters: (note, nom, prenom))
        db.commit()
        messagebox.showinfo( title: "Succès",  message: f"Note mise à jour pour {nom} {prenom} : {note}")
    open_window_module()
```

```python
window = Tk()
window.title("Modifier une note")
window.geometry("700x400")
window.resizable( width: False,  height: False)
frame = ttk.Frame(window, padding=40)
frame.pack(expand=True)
ttk.Label(frame, text="Modifier une note", font=("Segoe UI", 14, "bold")).grid(row=0, column=0, columnspan=2, pady=20)
ttk.Label(frame, text="Nom :").grid(row=1, column=0, pady=10, sticky="e")
ttk.Label(frame, text="Prénom :").grid(row=2, column=0, pady=10, sticky="e")
ttk.Label(frame, text="Nouvelle note :").grid(row=3, column=0, pady=10, sticky="e")
nom = StringVar()
prenom = StringVar()
note = DoubleVar()
ttk.Entry(frame, textvariable=nom, width=30).grid(row=1, column=1)
ttk.Entry(frame, textvariable=prenom, width=30).grid(row=2, column=1)
ttk.Entry(frame, textvariable=note, width=30).grid(row=3, column=1)
(ttk.Button(frame, text="Valider",
           command=lambda: [window.destroy(), verify(nom.get(), prenom.get(), note.get(), choix)])
 .grid(row=4, column=0, columnspan=2, pady=25))
window.mainloop()
```

cette fonction ouvre la fenêtre de modification d'une note.
Elle permet au professeur de saisir le nom, le prénom et la nouvelle note d'un étudiant.
En cliquant sur « Valider », la fonction appelle verify() pour vérifier si l'étudiant existe, puis mettre à jour la note correspondante (dans le module C ou Python) dans la base de données SQLite.
Un message de confirmation s'affiche ensuite pour signaler la réussite de la modification.

Page d’exportation des notes

Cette fenêtre permet au professeur d’exporter les notes des étudiants au format CSV.
Après avoir choisi le module (C ou Python), il clique sur « Exporter » pour sauvegarder automatiquement les notes depuis la base SQLite dans un fichier CSV.

```python
def open_window_export(choix):  1 usage
    window = Tk()
    window.geometry("350x300")
    window.title("Exportation des notes")
    frame = ttk.Frame(window, padding=40)
    frame.pack(expand=True)
    ttk.Label(frame, text="Exporter les notes au format CSV", font=("Segoe UI", 12, "bold")).pack(pady=20)
    ttk.Button(frame, text="Exporter", command=lambda: [window.destroy(), export(choix)], width=25).pack(pady=20)
    window.mainloop()
```

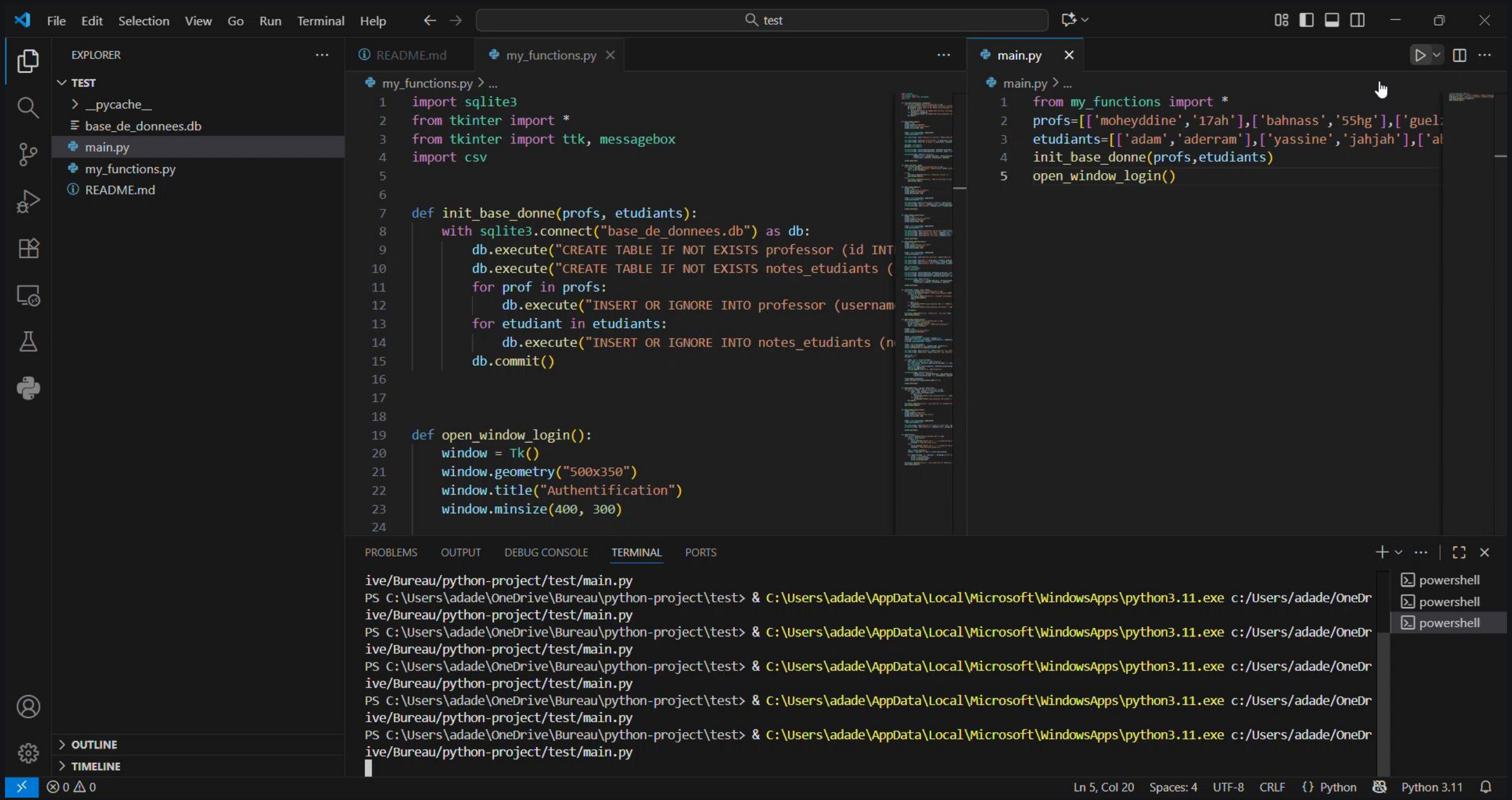
```python
def export(choix):  1 usage
    with sqlite3.connect("base_de_donnees.db") as conn:
        cursor = conn.cursor()
        if choix == 1:
            cursor.execute("SELECT nom || ' ' || prenom AS nom_complet, C AS note FROM notes_etudiants")
            filename = "exported_notes_C.csv"
        else:
            cursor.execute("SELECT nom || ' ' || prenom AS nom_complet, python AS note FROM notes_etudiants")
            filename = "exported_notes_python.csv"
        rows = cursor.fetchall()
        columns = [desc[0] for desc in cursor.description]
        with open(filename, "w", newline='', encoding="utf-8") as f:
            writer = csv.writer(f)
            writer.writerow(columns)
            writer.writerows(rows)
    messagebox.showinfo( title: "Succès",  message: f"Les notes ont été exportées vers '{filename}' !")
    open_window_module()
```

Cette fonction ouvre la fenêtre d’exportation des notes.
Elle affiche un bouton « Exporter » qui, lorsqu’il est cliqué, appelle la fonction export(choix).
Cette dernière récupère les notes du module sélectionné (C ou Python) depuis la base de données SQLite et les enregistre dans un fichier CSV.
Ainsi, cette fonction permet au professeur d’exporter facilement toutes les notes au format CSV.

ENSAMc

demo

Mustapha HAIN



Abderhaman Hajjam

Adam Aderraam

Yassine Jahjah

ENSAMc

Mustapha HAIN

Thank you

Abderhaman Hajjam

Adam Aderraam

Yassine Jahjah