<u>הנחיות למטלת פרויקט ב python</u> : מערכת ניהול ספרייה

תקציר

במטלה זו תידרשו לעצב וליישם **מערכת לניהול ספרייה** באמצעות Python. המערכת תאפשר ניהול מלאי ספרים, משתמשים, פעולות השאלה והחזרה, וכן תמיכה בחיפושים מתקדמים. המטרה היא להדגים יכולת תכנות מונחה עצמים (OOP), שימוש ב unit tests, שימוש ב **Design Patterns** ושמירה על עקרונות SOLID.

תיאור המערכת

מערכת הספרייה תאפשר לספרנים:

1. לנהל מלאי ספרים (הוספה, הסרה או עדכון).
2. לשמור נתוני ספרים בקבצים נפרדים (ספרים זמינים ומושאלים).
3. לעקוב אחר משתמשים והרשאותיהם לגישה למערכת.
4. לנהל רשימת המתנה לספרים פופולריים.
5. לאפשר חיפושים גמישים (למשל: לפי שם, מחבר, או קטגוריה).

ארכיטקטורת המערכת תתמוך במודולריות, הרחבה ועמידה בעקרונות **SOLID.** תשתמשו בתבניות עיצוב כגון:

- **Observer**: לעדכון מנויים בשינויים במערכת.
- **Strategy**: לתמיכה בחיפושים מרובים.
- **Decorator**: להוספת פונקציונליות דינמית לאובייקטים.
- **Iterator**: לניווט יעיל במאגרי הספרים.

משימות המטלה

1. ניהול ספרים

- יצירת מחלקה Book שתכלול פרטי ספרים (שם, מחבר, שנה, קטגוריה, עותקים ועוד).
- מעקב אחר מספר העותקים של כל ספר ועדכונם במקרה של השאלה או החזרה.

דרישות פונקציונליות:

- הוספת ספרים למאגר.
- הסרת ספרים מהמאגר.
- עדכון פרטי ספר ומעקב אחר זמינותו.
- סנכרון השינויים בקבצים: available_books.csv, loaned_books.csv, books.csv.
- כתיבת קובץ log מסוג txt. עם אינדיקציה להצלחה וכישלון של כל פעולה במטלה שנכתב שיש צורך לכתיבה לקובץ. שימו לב שהאינדיקציה היא על הצלחת הפעולה או שהתרחשה שגיאה. למשל האם החיפוש הצליח או שהחיפוש עצמו כשל עקב שגיאה בפתיחת הקובץ.

---

2. ניהול משתמשים

- יישום מערכת לניהול משתמשים שתומכת:
  - ברישום משתמשים עם סיסמאות מוצפנות.
  - באימות משתמשים בזמן התחברות.

דרישות פונקציונליות:

- שמירת פרטי משתמשים בקובץ users.csv.
- יישום אימות משתמשים באמצעות הצפנת סיסמאות.

ספריות אופציונליות להצפנה:

```
import hashlib
```

או

```
from werkzeug.security import generate_password_hash, check_password_hash
```

---

3. פעולות השאלה והחזרה

- הפעלת מנגנון להשאלת והחזרת ספרים.
- עדכון אוטומטי של סטטוס הספרים וסנכרונם בקבצים המתאימים.

דרישות פונקציונליות:

- מניעת השאלת ספר אם אין עותקים זמינים.
- הוספת משתמשים לרשימת המתנה לספרים פופולריים.

---

4. חיפוש והצגה

- יישום **Strategy Pattern** לחיפושים גמישים:
  - חיפוש לפי שם, מחבר, קטגוריה, ועוד.
  - תמיכה בהתאמות חלקיות.
- הצגת ספרים זמינים, מושאלים ופופולריים באמצעות ממשק משתמש גרפי (GUI).

---

5. עדכונים (Notifications)

- שימוש ב **Notifications** לעדכון מנויים על שינויים במערכת.
  - לדוגמה: עדכון משתמש כשהספר שחיכה לו הפך לזמין.

---

6. ממשק משתמש (GUI)

- יצירת ממשק משתמש גרפי לניהול הספרייה באמצעות Tkinter.
- הוספת כפתורים לפעולות:
  - הוספה / הסרת ספרים – הדפסה לקובץ log :
    - book added successfully/fail
    - book removed successfully/fail
  - חיפוש (שם ספר ושם סופר) והצגת ספרים (כלל הספרים, ספרים זמינים, ספרים מושאלים, ספרים לפי קטגוריה), כתיבה לקובץ log :
    - Search book "book name" by name completed successfully/fail
    - Search book "book name" by author name completed successfully/fail
    - Displayed all books successfully/fail
    - Displayed available books successfully/fail
    - Displayed borrowed books successfully/fail
    - Displayed book by category successfully/fail

    הערה- שימו לב שכל אחת מהאופציות היא כפתור בפני עצמה אך לא מופיעות בתפריט הראשי אלא במסך חדש שיפתח)
  - השאלת והחזרת ספרים, כתיבה לקובץ log :
    - book borrowed successfully/fail
    - book returned successfully/fail
  - יציאה – הדפסה לקובץ log :
    - log out successful/fail
  - מסך ראשי עם חיבור/הרשמה, הדפסה לקובץ log :
    - logged in successfully/fail
    - registered successfully/fail
  - הצגת ספרים פופולריים, כתיבה לקובץ log:
    - displayed successfully/fail
  - הצגת ספרים לפי ז'אנר(=קטגוריה), כתיבה לקובץ log :
    - displayed successfully/fail

עליכם להיצמד לשמות הכפתורים הבאים:

```
-  Add Book
-  Remove Book
-  Search Book
-  View Books
-  Lend Book
-  Return Book
-  Logout
-  Login
-  Register
```

```
-   Popular Books
```

מגבלות והנחיות יישום

1. **תבניות עיצוב**:
   - יישום **Strategy Pattern** - לחיפושים גמישים.
   - שימוש ב **Iterator Pattern** - לניווט יעיל.
   - שילוב **Decorator Pattern** - להוספת פונקציונליות דינמית.
2. **עקרונות SOLID**:
   - שמירה על עקרון האחריות היחידה, פתיחות / סגירות והיפוך תלות.
3. **ניהול קבצים**:
   - סנכרון נתוני ספרים בקבצים:
     - books.csv

   הערה: במידה וברצונכם להוסיף קבצים כמו available_books.csv loaned_books.csv הרשות בידכם

4. **ניהול שגיאות**:
   - הצגת הודעות שגיאה ברורות להזנת נתונים שגויים.
   - טיפול בשגיאות בקבצים בצורה חסינה.

תוצרי המטלה

1. **קוד הפרויקט**:
   - כל קבצי ה-Python כולל ממשק המשתמש והלוגיקה העסקית.
2. **תיעוד**:
   - README.md שיכלול:
     - הוראות להרצת הפרויקט.
     - תיאור תכונות המערכת.
     - תבניות העיצוב שהוטמעו.
3. **בדיקות**:
   - בדיקות יחידה לכל הפונקציות הקריטיות.
4. **הצגה**:
   - וידאו קצר המסביר את המערכת (אופציונלי אך מומלץ).

קריטריונים להערכה

משקל	תיאור	קריטריון
40%	עמידה בדרישות הפונקציונליות של המערכת	פונקציונליות
20%	יישום נכון ויעיל של תבניות העיצוב הנדרשות	שימוש בתבניות עיצוב
20%	שמירה על עקרונות SOLID, קריאות ותחזוקתיות	איכות הקוד
10%	שימושיות ושלמות ממשק המשתמש	ממשק משתמש (GUI)
10%	בהירות התיעוד ושלמות בדיקות היחידה	תיעוד ובדיקות

התחלה

- הגשה של הפרוייקט תהיה בזיפ במודל
- קראו את הנתונים בקובץ books.csv על מנת להבין את מבנה הנתונים.
- התחילו ביישום מחלקת Book ויצירתה באמצעות פאקטורי.
- המשיכו להטמעת התבניות הנדרשות בשלבים הבאים.

לשאלות, ניתן לפנות בפורום במודל, בהצלחה!