FEEDBACKS

(Feedbacks were provided from Justine, Aman, Alexis, and Nereus.

1. *Your `show_books` function now displays data in a much cleaner and more organized way, which really enhances readability.*

2. *The formatting improvements make the table look more professional and easier to scan quickly.*

3. *I appreciate how you've added input validation for the publication date and number of pages, as it helps ensure data accuracy.*

4. *This validation also prevents potential errors or inconsistencies in the database entries.*

5. *The use of try-except blocks around database setup and data retrieval makes your program more robust and error-resistant.*

6. *It's great to see you handling potential exceptions gracefully instead of letting the program crash unexpectedly.*

7. *Your code is now more modular with clear functions like `add_book`, `show_books`, and `setup_database`, which improves readability.*

8. *Modular code like this makes it easier to maintain and expand in the future.*

9. *The user interface with easygui pop-ups is very user-friendly and makes the program accessible for people unfamiliar with command-line tools.*

10. *The prompts are clear, guiding the user effectively through each step.*

11. *Your feedback messages after each action help users know that their inputs were successful, improving the overall experience.*

12. *The program now handles common user errors better, such as invalid inputs, which makes it more reliable.*

13. *I noticed you've organized your code logically, which makes it easier to follow and understand your thought process.*

14. *The code comments you've added help clarify complex sections, which is helpful for future reference or for others reading your code.*

15. *Your use of string formatting to align the table columns shows attention to detail and makes the output look neat.*

16. *To improve further, you might consider dynamically adjusting the column widths based on the data length for even better alignment.*

17. *You could also add functionality to allow editing or deleting existing entries, which would make the program more versatile.*

18. *Storing previous data entries temporarily could help users compare or verify their inputs easily.*

19. *Implementing a search or filter feature to find books by author or publication year could enhance usability.*

20. *Considering a way to export the data to a file might be a nice addition for users who want to save or share their data.*

21. *Adding comments or docstrings to your functions would improve code documentation and help others understand your logic faster.*

22. *You could consider adding some input prompts or instructions at the beginning to guide new users through the process.*

23. *The overall flow of your program feels smooth, and I can see you're making steady progress in your coding skills.*

24. *Your approach to handling database connections is good, but you might want to think about closing the connection properly after operations are done.*

25. *Including some basic unit tests or validation checks could help ensure your functions work correctly as you add new features.*

26. *You might want to explore using classes if your project becomes more complex - it can help organize related data and functions better.*

27. *The readability of your code is quite good, but breaking some larger functions into smaller ones could make it even clearer.*

28. *Your enthusiasm for improving and refining your project is evident, and that's a great attitude to have.*

29. *Overall, you've done a fantastic job so far, and your attention to detail and user experience is really paying off. Nice.*

FINAL SELF-REVIEW

Based on the feedback, I made several key improvements to my project. I enhanced the display of my book list by formatting it into a clean, organized table, which makes it much easier for users to read and interpret the data quickly. I also added input validation for critical fields like publication date and number of pages, making sure that the data entered is accurate and consistent, which helps maintain the integrity of my database. To make my program more stable and user-friendly, I incorporated try-except blocks around database operations. This allows my program to handle errors gracefully instead of crashing unexpectedly, providing a smoother experience for users. I also broke down my code into smaller, modular functions such as `add_book, show_books, and setup_database`*, improving readability and making it easier to maintain and expand in the future. Additionally, I improved the user interface by utilizing easygui pop-ups for inputs and messages, making interactions more intuitive and accessible. I also made sure prompts and success messages guide users effectively through each step, which enhances overall usability. I paid close attention to formatting details, like aligning columns properly for a professional look. I also thought about the future enhancements, such as dynamically adjusting column widths, adding editing or deleting features, and enabling data export options, which will make my project more versatile and user-focused. Overall, these changes reflect my commitment to creating a more reliable, organized, and user-friendly application, and I am applying feedback thoughtfully to improve my coding skills and project quality.*