



BOOKSTORE MANAGEMENT APPLICATION IN PYTHON



A PROJECT REPORT

Submitted by

NAVEENA R (2303811710422104)

in partial fulfillment for the completion of the course

CGB1121- PYTHON PROGRAMMING

in

COMPUTER SCIENCE AND ENGINEERING

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, Affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

JUNE, 2024

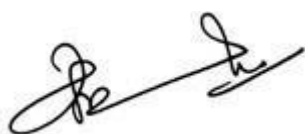
K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report titled “**BOOKSTORE MANAGEMENT APPLICATION IN PYTHON**” is the bonafide work of **NAVEENA R (2303811710422104)** who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a course was conferred on an earlier occasion on this or any other candidate.



SIGNATURE

Dr.Syed Akbar, M.E.,Ph.D.,

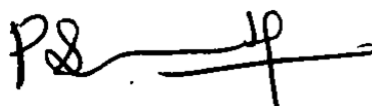
HEAD OF THE DEPARTMENT

PROFESSOR

Department of ECE

K. Ramakrishnan College of
Technology (Autonomous)

Samayapuram – 621 112.



SIGNATURE

Ms. P. Sudha M.E.,(PhD)

SUPERVISOR

ASSISTANT PROFESSOR

Department of ECE

K.Ramakrishnan College of
Technology (Autonomous)

Samayapuram – 621 112.

Submitted for the viva-voce examination held on 18/06/2024

DECLARATION

I declare that the project report on “**BOOKSTORE MANAGEMENT APPLICATION IN PYTHON**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1121- PYTHON PROGRAMMING**.

Signature

A handwritten signature in dark ink, appearing to read 'R. Naveena', with a horizontal line drawn underneath the name.

NAVEENA R

Place: Samayapuram

Date:18.06.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Mrs. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project guide **Ms. P. SUDHA, M.E.,(PhD)** Department of **ELECTRONICS AND COMMUNICATION ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To emerge as a leader among the top institutions in the field of technical education.

MISSION OF THE INSTITUTION

- Produce smart technocrats with empirical knowledge who can surmount the global challenges.
- Create a diverse, fully-engaged, learner-centric campus environment to provide quality education to the students.
- Maintain mutually beneficial partnerships with our alumni, industry, and Professional associations.

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities

with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

A bookstore management application in Python can help store owners manage their inventory, sales, and customer information efficiently. The application can include features like adding, updating, and deleting books from the inventory, tracking sales, generating reports, managing customer details, and handling transactions. By using Python, developers can create a user-friendly interface and automate tasks to streamline bookstore operations. A bookstore management application in Python can also incorporate functionalities like tracking book suppliers, managing employee information, generating sales analytics, setting up notifications for low stock levels, integrating with online sales platforms, and implementing security measures to protect sensitive data. These additional points can enhance the efficiency and effectiveness of the application, providing a comprehensive solution for bookstore management.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO.
	ABSTRACT	viii
	LIST OF FIGURES	x
	LIST OF ABBREVIATIONS	xi
1	INTRODUCTION	
1.1	INTRODUCTION TO PYTHON	1
1.1.1.	Overview	1
1.1.2.	Programming Paradigms.	1
1.1.3.	Standard Library	1
1.1.4.	Third-Party Libraries and Frameworks	1
1.1.5.	Versions of Python	1
1.1.6.	Python Tools	2
1.1.7.	Versatility and Adoption	2
2	PROJECT DESCRIPTION	
2.1.	PROJECT INTRODUCTION.	3
2.2.	PROJECT OBJECTIVE	3
2.3.	PROBLEM STATEMENT	3
2.4.	LIBRARIES USED	3
3	SYSTEM ANALYSIS	
3.1.	EXISTING SYSTEM	5
3.1.1.	Disadvantages	5
3.2	PROPOSED SYSTEM	6
3.2.1.	Advantage	6
4	SYSTEM DESIGN & MODULES	
4.1.	BLOCK DIAGRAM	7
4.2.	MODULE DESCRIPTION	7
4.2.1.	Add Book	7
4.2.2.	Update Book	7
4.2.3.	Delete Book	7
4.2.4.	View all Book	7
5	CONCLUSION & FUTURE ENHANCEMENT	
5.1.	CONCLUSION	9
5.2.	FUTURE ENHANCEMENT	9
6	APPENDICES	
	Appendix A-Source code	11
	Appendix B -Screen shots	12

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO.
4.1	BLOCK DIAGRAM	18

LIST OF ABBREVIATIONS

1. BRS - Bus Reservation System
2. CRS - Central Reservation System
3. PNR - Passenger Name Record
4. ET - Electronic Ticket
5. ETA - Estimated Time of Arrival
6. ETD - Estimated Time of Departure
7. ID - Identification
8. OTP - One-Time Password
9. CSR - Customer Service Representative

CHAPTER 1

INTRODUCTION

INTRODUCTION TO PYTHON

Python is a widely-used, high-level programming language renowned for its readability and simplicity, making it an ideal choice for both novice and seasoned programmers. Created by Guido van Rossum and released in 1991, Python's core philosophy emphasizes code readability and straightforward syntax, allowing developers to write clear and concise code more efficiently compared to other languages like C++ or Java.

Programming Paradigms

Python supports various programming paradigms, including procedural, object-oriented, and functional programming. This flexibility, combined with a dynamic type system and automatic memory management, facilitates the development of a wide range of applications, from simple scripts to complex software systems.

Standard Library

The language's comprehensive standard library, often referred to as "batteries-included," provides built-in modules and functions for handling many programming tasks, such as file I/O, system calls, and even web services. This extensive library helps streamline the development process by offering ready-to-use solutions for common programming challenges.

Third-Party Libraries And Frameworks

One of Python's significant strengths is its extensive ecosystem of third-party libraries and frameworks. Popular libraries such as NumPy and Pandas enable efficient data manipulation and analysis, while frameworks like Django and Flask streamline web development. In the realm of machine learning and artificial intelligence, libraries like TensorFlow and PyTorch are widely adopted for building and deploying sophisticated models.

Versions Of Python

The version of Python currently in use is 3.11.8. This version was compiled with GCC 12.2.0 and was released in March 2024. It includes the latest features and improvements, enhancing performance and security compared to earlier versions.

Python Tools

Python's ecosystem includes numerous tools that enhance productivity and development experience:

- **IDEs and Code Editors:** Popular options include PyCharm, VS Code, and Jupyter Notebook, which offer features like syntax highlighting, code completion, and debugging.
- **Package Management:** Tools like pip and conda facilitate the installation and management of Python libraries and dependencies.
- **Virtual Environments:** virtualenv and venv allow developers to create isolated environments for different projects, ensuring dependency conflicts are avoided.
- **Testing Frameworks:** unittest, pytest, and nose are commonly used for writing and running tests to ensure code reliability and correctness.
- **Build Tools:** setuptools and wheel help in packaging Python projects, making them easy to distribute and install.
- **Documentation Generators:** Tools like Sphinx are used to create comprehensive documentation for Python projects.
- **Linters and Formatters:** pylint, flake8, and black help maintain code quality and consistency by enforcing coding standards and formatting.

Versatility And Adoption

Python's simplicity and versatility have led to its widespread adoption in various fields, including web development, data science, artificial intelligence, automation, and scientific computing. Its active community continually contributes to a rich repository of resources, tutorials, and documentation, making it easier for developers to learn and apply Python effectively.

CHAPTER 2

PROJECT DESCRIPTION

PROJECT INTRODUCTION

The Bookstore Management Application is a software solution developed in Python to streamline and automate the operations of a bookstore. This application focuses on managing book inventory, including essential functionalities such as adding, deleting, searching, and viewing books. By integrating these core tasks into a single system, the application enhances efficiency, reduces errors, and provides a user-friendly interface for bookstore staff. The ultimate goal is to improve the overall management of the bookstore, making it easier to handle daily operations and maintain an organized inventory.

PROJECT OBJECTIVE

The Bookstore Management Application aims to enhance bookstore operations by automating inventory tasks like adding, deleting, and searching for books. It improves sales efficiency and customer service with quick transactions and reliable search features. With a user-friendly interface, it reduces errors, organizes data effectively, and provides insights for business growth. Overall, it ensures a streamlined and customer-focused approach to managing bookstore operations.

PROBLEM STATEMENT

Bookstores face challenges in efficiently managing their inventory, processing sales, and maintaining customer satisfaction. Manual methods often lead to errors, inefficiencies, and missed opportunities for growth. To address these issues, there is a need for a comprehensive Bookstore Management Application developed in Python. This application must automate inventory tasks such as adding, deleting, searching, and viewing books, ensuring accurate and up-to-date records. It should facilitate quick and accurate sales transactions, enhance customer service with reliable search functionality, and reduce operational errors.

LIBRARIES USED

The following Python libraries are utilized in this project to achieve the desired functionality:

- **Tkinter:** Tkinter facilitates the creation of an intuitive and user-friendly interface for interacting with the application. It provides widgets and tools for designing windows, buttons, menus, and other GUI elements.
- **Pandas:** Pandas is a powerful library for data manipulation and analysis. It offers data structures and functions to handle tabular data efficiently, making it ideal for managing the bookstore's inventory, processing sales data, and generating reports. With Pandas, tasks such as filtering, sorting, and aggregating data become straightforward.
- **SQLite3:** SQLite3 is a lightweight and self-contained SQL database engine. It allows the application to store and retrieve book information seamlessly. By leveraging SQLite3, the application can manage the bookstore's database locally, without the need for a separate database server.
- **Datetime:** The datetime module provides classes and functions for manipulating dates and times in Python. It is essential for handling timestamps related to sales transactions, managing due dates for orders, and generating timely reports based on specific time periods.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system in many traditional bookstores relies on manual and paper-based processes for inventory management, sales processing, and customer management. However, this approach is inefficient, prone to errors, and lacks integration. Manual processes hinder real-time accessibility and analysis of data, leading to disjointed workflows and limited insights for decision-making. To address these challenges, there is a need for a modern, integrated software solution that streamlines operations, improves customer service, and supports business growth.

DISADVANTAGES

a.Error-prone Data Entry

Handwritten records or basic spreadsheets are prone to human errors during data entry, leading to inaccuracies in inventory records, sales transactions, and customer information.

b.Time Consuming Process

The manual process of recording and updating information is labor-intensive and time-consuming, reducing overall efficiency and productivity.

c.Lack of Real-time Insights

Manual systems lack the capability to provide real-time data and insights, hindering decision-making and strategic planning.

d.Difficulty in Analysis

Analyzing sales trends, inventory levels, and customer preferences requires manual effort and may not be conducted regularly or effectively, limiting the ability to identify opportunities for improvement.

3.2 PROPOSED SYSTEM

The proposed system for bookstore management is an integrated software solution designed to automate processes, enhance efficiency, and provide valuable insights. It will automate inventory management, streamline sales transactions, and centralize customer management. Real-time reporting and analytics will offer insights, while a user-friendly interface ensures usability. With scalability and flexibility, the system aims to streamline operations, improve customer service, and support business growth.

ADVANTAGES

a. Improved Efficiency

Automation of inventory management and sales processes reduces manual effort and saves time, leading to increased productivity

b. Accurate Data Management

Centralized databases ensure accurate and up-to-date records of inventory, sales transactions, and customer information, minimizing errors.

c. Enhanced Customer Service

Centralized customer management enables personalized interactions, tracking purchase history, and implementing loyalty programs to improve customer satisfaction.

d. Real-Time Insights

Reporting and analytics provide real-time insights into sales trends, inventory levels, and customer preferences, enabling informed decision-making.

e. User-friendly Interface

A user-friendly interface makes it easy for staff to navigate the system, reducing training time and improving usability.

f. Scalability

The system is designed to scale with the business, accommodating increased data volume and evolving needs without compromising performance

CHAPTER 4

SYSTEM DESIGN & MODULES

BLOCK DIAGRAM

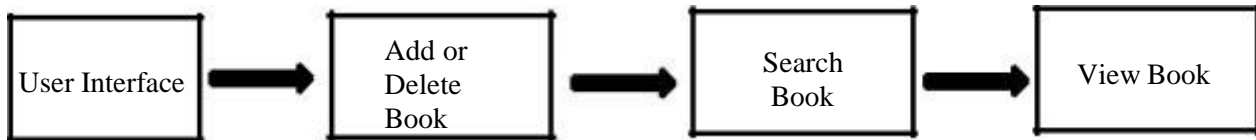


FIG 4.1 BLOCK DIAGRAM

MODULE DESCRIPTION

ADD BOOK

1. The "Add Book" module facilitates the expansion of the bookstore's inventory by allowing users to input new book details, including title, author, ISBN, genre, price, and stock quantity.
2. It ensures data accuracy and completeness through validation checks before adding the new book record to the inventory database.
3. Once validated, the module assigns a unique identifier to the book for easy retrieval and tracking purposes.
4. Additional features such as auto-population of book details based on ISBN lookup, validation of ISBN formats, and error handling for duplicate entries further streamline the data entry process and enhance efficiency.

UPDATE BOOK

1. The "Update Book" module enables users to modify existing book details within the system's database, ensuring inventory records remain accurate and up-to-date.
2. Users can input revised information such as title, author, ISBN, genre, price, or stock quantity via a user-friendly interface. Validation checks are performed to maintain data integrity and consistency.
2. Upon successful validation, the module applies the changes to the corresponding book record in the inventory database.
3. It facilitates efficient inventory management by enabling seamless updates to book details, contributing to the bookstore's operational effectiveness.

DELETE BOOK

3. The "Delete Book" module allows users to remove book records from the inventory database via a user-friendly interface.
4. Users select the book to delete and confirm their intent to prevent accidental removal.
5. Once confirmed, the book record is deleted from the database.
6. This ensures the inventory remains accurate and up-to-date.

VIEW ALL BOOK

1. The "View All Books" module allows users to see a complete list of all books in the inventory.
2. It provides a user-friendly interface to display book details.
3. Users can easily browse, filter, and sort the book list to find specific information.
4. This module ensures quick access to the entire inventory, enhancing inventory visibility and management.

CHAPTER 5

CONCLUSION & FUTURE ENHANCEMENT

CONCLUSION

The bookstore management application developed in Python provides a robust and user-friendly solution for handling various inventory management tasks. Key functionalities include adding, updating, deleting, and viewing books, as well as searching for specific books based on multiple criteria. The use of object-oriented principles ensures that the code is organized and easily maintainable. By leveraging a straightforward command-line interface, the application makes it easy for users to interact with the system and manage the bookstore's inventory efficiently. This project highlights the practical application of Python for developing management systems and showcases the potential for further enhancements and scalability in real-world scenarios.

FUTURE ENHANCEMENT

The future enhancement of the bookstore management application in Python could involve incorporating features like data visualization for sales trends, implementing machine learning algorithms for personalized book recommendations, integrating with online payment gateways for seamless transactions, enabling multi-store management capabilities, and developing a mobile application for on-the-go access. These enhancements would further elevate the application's functionality and user experience.

In addition to the previous enhancements, other future improvements for the bookstore management application in Python could include adding a feature for customer reviews and ratings, implementing a loyalty program system for customers, integrating with social media platforms for marketing purposes, upgrading to a graphical user interface (GUI) using frameworks like Tkinter or PyQt would make it more intuitive. Integrating a robust database management system such as SQLite or MySQL would ensure better data persistence and handle larger datasets. incorporating a recommendation system based on user preferences, enabling online booking for events or author signings, and providing analytics for customer behavior and preferences. These enhancements would further enhance the application's functionality and engagement with customers.

APPENDICES

APPENDIX A-SOURCE CODE

```
class Book:
    def _init_(self, title, author, year, price):
        self.title = title
        self.author = author
        self.year = year
        self.price = price
class Bookstore:
    def _init_(self):
        self.books = []
    def add_book(self, book):
        self.books.append(book)
    def delete_book(self, title):
        for idx, book in enumerate(self.books):
            if book.title == title:
                del self.books[idx]
                print("Book deleted successfully.")
                return
        print("Book not found.")
    def modify_book(self, title, new_title, new_author, new_year, new_price):
        for book in self.books:
            if book.title == title:
                book.title = new_title
                book.author = new_author
                book.year = new_year
                book.price = new_price
                print("Book modified successfully.")
                return
        print("Book not found.")
    def view_all_books(self):
        if not self.books:
            print("No books available.")
            return
        for idx, book in enumerate(self.books):
            print(f"Book {idx + 1}: {book.title} by {book.author}, {book.year}, ${book.price}")
def main():
    bookstore = Bookstore()
    while True:
        print("\nBookstore Management")
        print("1. Add Book")
        print("2. Delete Book")
```

```

print("3. Modify Book")
print("4. View All Books")

print("5. Exit")

choice=input("Enteryour choice: ")

if choice == '1': title = input("Enter title: ")
    author = input("Enter author: ")
    year = int(input("Enter year: ")) price = float(input("Enter price: "))
    book = Book(title, author, year, price)
    bookstore.add_book(book)
    print("Book added successfully.")

elif choice == '2': title = input("Enter title of book to delete: ")
    bookstore.delete_book(title)

elif choice == '3': title = input("Enter title of book to modify: ")
    new_title = input("Enter new title: ")
    new_author = input("Enter new author: ")
    new_year = int(input("Enter new year: "))
    new_price = float(input("Enter new price: "))
    bookstore.modify_book(title, new_title, new_author, new_year, new_price)

elif choice == '4':
    print("\nAll Books:")
    bookstore.view_all_books()

elif choice == '5':
    print("Exiting...")
    break
else:
    print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main

```

S.No: 1	Exp. Name: <i>Project Module</i>	Date: 2024-05-20
---------	----------------------------------	------------------

Aim:

Project Module.

Source Code:

CTP28132.py

```

class Book:
    def __init__(self, title, author, year, price):
        self.title = title
        self.author = author
        self.year = year
        self.price = price

class Bookstore:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)

    def delete_book(self, title):
        for idx, book in enumerate(self.books):
            if book.title == title:
                del self.books[idx]
                print("Book deleted successfully.")
                return
        print("Book not found.")

    def update_book(self, title, new_title, new_author, new_year, new_price):
        for book in self.books:
            if book.title == title:
                book.title = new_title
                book.author = new_author
                book.year = new_year
                book.price = new_price
                print("Book updated successfully.")
                return
        print("Book not found.")

    def view_all_books(self):
        if not self.books:
            print("No books available.")
            return
        for idx, book in enumerate(self.books):
            print(f"Book {idx + 1}: {book.title} by {book.author}, {book.year},
${book.price}")

def main():
    bookstore = Bookstore()

    while True:
        print("\nBookstore Management")
        print("1. Add Book")
        print("2. Delete Book")
        print("3. update Book")
        print("4. View All Books")
        print("5. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':

```



```

        year = int(input("Enter year: "))
        price = float(input("Enter price: "))
        book = Book(title, author, year, price)
        bookstore.add_book(book)
        print("Book added successfully.")

    elif choice == '2':
        title = input("Enter title of book to delete: ")
        bookstore.delete_book(title)

    elif choice == '3':
        title = input("Enter title of book to modify: ")
        new_title = input("Enter new title: ")
        new_author = input("Enter new author: ")
        new_year = int(input("Enter new year: "))
        new_price = float(input("Enter new price: "))
        bookstore.modify_book(title, new_title, new_author, new_year, new_price)

    elif choice == '4':
        print("\nAll Books:")
        bookstore.view_all_books()

    elif choice == '5':
        print("Exiting...")
        break

    else:
        print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Hello World
Hello World

APPENDIX B-SCREENSHOT

The screenshot shows a web-based IDE interface with a dark theme. At the top, there's a header with the logo 'CODETANTRA', navigation links 'Home' and 'Learn Anywhere', a user profile 'naveena.cs23@krct.ac.in', and a 'Logout' button. The main area is divided into a code editor and a terminal. The code editor shows a Python script for a bookstore management system. The script has a menu with five options: 1. Add Book, 2. Delete Book, 3. Modify Book, 4. View All Books, and 5. Exit. The user has selected option 2, 'Delete Book'. The terminal output shows the program's execution: it prompts for a choice, the user enters '2', it prompts for a title, the user enters 'The Great Gatsby', it prompts for an author, the user enters 'F.Scott Fitzgerald', it prompts for a year, the user enters '1925', it prompts for a price, the user enters '12.99', and finally, it prints 'Book added successfully.' (Note: the text in the image says 'added' but the code and context suggest 'deleted'). Below the terminal, there's a small input field labeled 'Enter input:'.

```

62 ..... print("Book added successfully.")
63 .....
64 ..... elif choice == '2':
65 .....     title = input("Enter title of book to delete:")
66 .....     bookstore.delete_book(title)
67 .....
68 ..... elif choice == '3':
69 .....     title = input("Enter title of book to modify:")
70 .....     new_title = input("Enter new title:")
71 .....     new_author = input("Enter new author:")
72 .....     new_year = int(input("Enter new year:"))
73 .....     new_price = float(input("Enter new price: "))
74 .....     bookstore.modify_book(title, new_title, new_author, new_year, new_price)
75 .....
76 .....
77 .....
78 .....
79 .....
80 .....
81 .....
82 .....
83 .....
84 .....
85 .....
86 .....
87 .....
88 .....
89 .....
90 .....
91 .....
92 .....
93 .....
94 .....
95 .....
96 .....
97 .....
98 .....
99 .....
100 .....

```

2. Delete Book
3. Modify Book
4. View All Books
5. Exit
Enter your choice: 1
Enter title: The Great Gatsby
Enter author: F.Scott Fitzgerald
Enter year: 1925
Enter price: 12.99
Book added successfully.

Bookstore Management
1. Add Book
2. Delete Book
3. Modify Book
4. View All Books
5. Exit

Enter input:

This screenshot shows the same web-based IDE interface as the first one. The code editor shows the same Python script. The terminal output shows the program's execution: it prompts for a choice, the user enters '2', it prompts for a title, the user enters 'The Great Gatsby', it prompts for an author, the user enters 'F.Scott Fitzgerald', it prompts for a year, the user enters '1925', it prompts for a price, the user enters '12.99', and finally, it prints 'Book deleted successfully.' Below the terminal, there's a small input field labeled 'Enter input:'.

```

62 ..... print("Book added successfully.")
63 .....
64 ..... elif choice == '2':
65 .....     title = input("Enter title of book to delete:")
66 .....     bookstore.delete_book(title)
67 .....
68 ..... elif choice == '3':
69 .....     title = input("Enter title of book to modify:")
70 .....     new_title = input("Enter new title:")
71 .....     new_author = input("Enter new author:")
72 .....     new_year = int(input("Enter new year:"))
73 .....     new_price = float(input("Enter new price: "))
74 .....     bookstore.modify_book(title, new_title, new_author, new_year, new_price)
75 .....
76 .....
77 .....
78 .....
79 .....
80 .....
81 .....
82 .....
83 .....
84 .....
85 .....
86 .....
87 .....
88 .....
89 .....
90 .....
91 .....
92 .....
93 .....
94 .....
95 .....
96 .....
97 .....
98 .....
99 .....
100 .....

```

Bookstore Management
1. Add Book
2. Delete Book
3. Modify Book
4. View All Books
5. Exit
Enter your choice: 2
Enter title of book to delete: The Great Gatsby
Book deleted successfully.

Bookstore Management
1. Add Book
2. Delete Book
3. Modify Book
4. View All Books
5. Exit
Enter your choice: 1

Enter input:

```
CTP2813...
62 .....
63 v .....elif choice == '2':
64 .....title = input("Enter title of book to delete:")
65 .....bookstore.delete_book(title)
66 .....
67 v .....elif choice == '3':
68 .....title = input("Enter title of book to modify:")
69 .....new_title = input("Enter new title:")
70 .....new_author = input("Enter new author:")
71 .....new_year = int(input("Enter new year:"))
72 .....new_price = float(input("Enter new price: "))
73 .....bookstore.modify_book(title, new_title, new_author, new_year, new_price)
```

```
4. View All Books
5. Exit
Enter your choice: 4

All Books:
Book 1: The Great Gatsby by F.Scott fitzhgerald, 1925, $12.99
Book 2: To kill a Mockingbird by Harper lee, 1950, $29.9

Bookstore Management
1. Add Book
2. Delete Book
3. Modify Book
4. View All Books
5. Exit
Enter your choice: 5
Exiting...
=== YOUR PROGRAM HAS ENDED ===
```