TITLE OF PROJECT REPORT

# ONLINE BOOK STORE

*Submitted by*

**Tanisha Jain(24MCI10047)**

**Jaspreet Kaur(24MCI10062)**

*In partial fulfilment for the award of the degree of*

**Masters of computers application (AI&ML)**

**IN**

BRANCH OF STUDY

AI&ML

CU CHANDIGARH UNIVERSITY

CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

**CHANDIGARH UNIVERSITY**

MONTH&YEAR

NOVEMBER-2024-25

| | |
|---|---|
| **Name:** Tanisha Jain & Jaspreet Kaur | **Subject Code:** 24CAH |
| **Branch:** MCA(AI&ML) | **Date of Performance:** 1-11-2024 |
| **UID:** 24MCI10047&24MCI10062 | **Semester:** $1^{st}$ |
| **Subject Name:** DAA | **Section/ Group: 1/B** |

## Project- Online book Store

**Aim:**

The aim of this project is to implement a basic online bookstore application using Python and Tkinter with core functionalities like book browsing, search, cart management, and checkout. By leveraging concepts from the Design and Analysis of Algorithms (DAA), the project demonstrates the application of:

- **Algorithmic Techniques**: Sorting and searching techniques (merge sort and binary search) are integrated to enhance user experience by allowing quick and organized book listings and searches.
- **GUI Development**: Tkinter is used to build an interactive and user-friendly interface that enables users to interact with the bookstore.

**Task to be done:**

**1. Setup and Basic UI Design**

- Set up the main Tkinter window.
- Design the layout for the bookstore interface, including sections for available books, search bar, cart, and checkout.

**2.Implement Data Structures**

- Define data structures (list of dictionaries) to store book information, including title, author, price, and stock availability.

### 3. Sorting and Searching

- Implement Merge Sort to display books alphabetically.
- Implement Binary Search for quick book lookups by title.

### 4. Display Books in Listbox

- Create a listbox widget to display the books.
- Write a function to populate the listbox with book details.

### 5. Functionality

- Implement a search bar to enter book titles.
- Set up a search button to trigger binary search and display the result.

### 6. Add to Cart

- Allow users to select a book and add it to a cart.
- Update stock availability each time a book is added to the cart.

### 7. Cart and Checkout

- Create a cart list to store selected books.
- Implement a checkout button to calculate the total price of items in the cart.

- Display a summary message of the total amount upon checkout.

## CODING

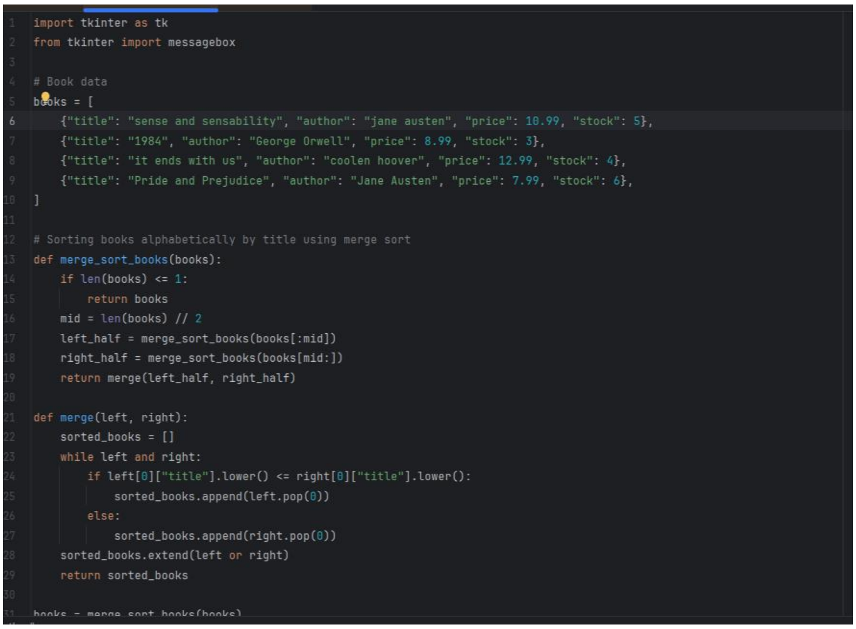
```
import tkinter as tk

from tkinter import messagebox



# Book data

books = [
    {"title": "The Great Gatsby", "author": "F. Scott Fitzgerald", "price": 10.99,
"stock": 5},

    {"title": "1984", "author": "George Orwell", "price": 8.99, "stock": 3},

    {"title": "To Kill a Mockingbird", "author": "Harper Lee", "price": 12.99,
"stock": 4},

    {"title": "Pride and Prejudice", "author": "Jane Austen", "price": 7.99,
"stock": 6},

]


# Sorting books alphabetically by title using merge sort

def merge_sort_books(books):

    if len(books) <= 1:

        return books



 mid = len(books) // 2

    left_half = merge_sort_books(books[:mid])

    right_half = merge_sort_books(books[mid:])

    return merge(left_half, right_half)


def merge(left, right):

    sorted_books = []

    while left and right:

        if left[0]["title"].lower() <= right[0]["title"].lower():

            sorted_books.append(left.pop(0))
        else:
            sorted_books.append(right.pop(0))
    sorted_books.extend(left or right)
    return sorted_books

books = merge_sort_books(books)



# Search function (binary search)
def binary_search(books, title):
    low, high = 0, len(books) - 1
    while low <= high:
        mid = (low + high) // 2
        if books[mid]["title"].lower() == title.lower():
            return books[mid]
        elif books[mid]["title"].lower() < title.lower():
            low = mid + 1
        else:
            high = mid - 1
    return None

# Initialize main Tkinter window
root = tk.Tk()
```

```
root.title("Online Bookstore")

# Cart to store selected books
cart = []

# Display books in store
def display_books():
    listbox.delete(0, tk.END)
    for book in books:
        listbox.insert(tk.END, f"{book['title']} - ${book['price']} (Stock: {book['stock']})")

# Add book to cart
def add_to_cart():
    selected_book = listbox.get(tk.ACTIVE)



    if not selected_book:
        messagebox.showinfo("Info", "Please select a book to add.")
        return

    title = selected_book.split(" - ")[0]
    book = binary_search(books, title)
    if book and book["stock"] > 0:
        book["stock"] -= 1
        cart.append(book)
        messagebox.showinfo("Cart", f"Added {book['title']} to the cart.")
        display_books()
    else:
        messagebox.showwarning("Warning", "Book out of stock!")

# Search for a book by title

def search_book():
    title = search_entry.get()
    book = binary_search(books, title)
    if book:
        listbox.delete(0, tk.END)
        listbox.insert(tk.END, f"{book['title']} - ${book['price']} (Stock: {book['stock']})")
    else:
        messagebox.showinfo("Search", "Book not found.")

# Checkout function to display cart items
```

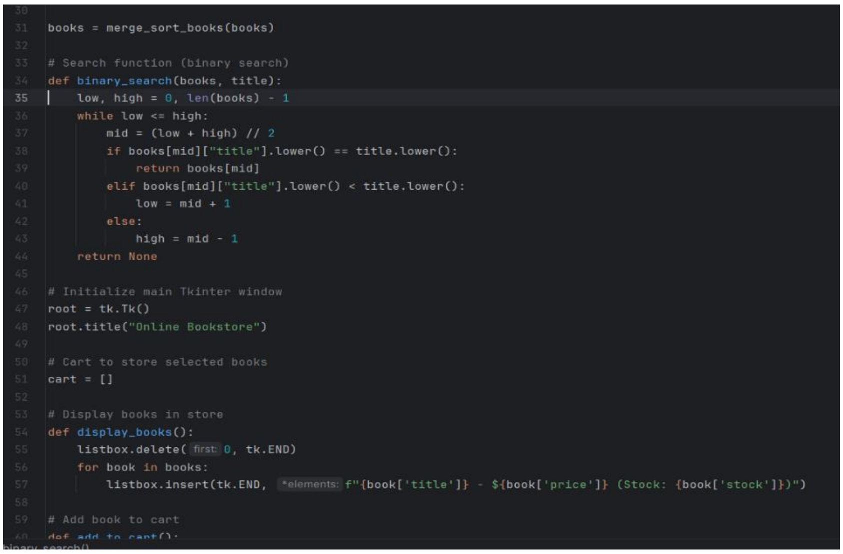
```
def checkout():
    if not cart:
        messagebox.showinfo("Cart", "Your cart is empty.")

  return
    total_price = sum(item["price"] for item in cart)
    messagebox.showinfo("Checkout", f"Total Price: ${total_price:.2f}")
    cart.clear()

# UI Elements
frame = tk.Frame(root)
frame.pack(pady=10)

# Search bar
search_entry = tk.Entry(frame, width=30)
search_entry.grid(row=0, column=0, padx=5)
search_button = tk.Button(frame, text="Search", command=search_book)
search_button.grid(row=0, column=1)




# Book list display
```

```
listbox = tk.Listbox(root, width=50, height=10)

listbox.pack(pady=10)

# Add to Cart button

add_button = tk.Button(root, text="Add to Cart", command=add_to_cart)

add_button.pack(pady=5)

# Checkout button

checkout_button = tk.Button(root, text="Checkout", command=checkout)

checkout_button.pack(pady=5)

# Load initial book display

display_books()


# Run the main loop

root.mainloop()
```

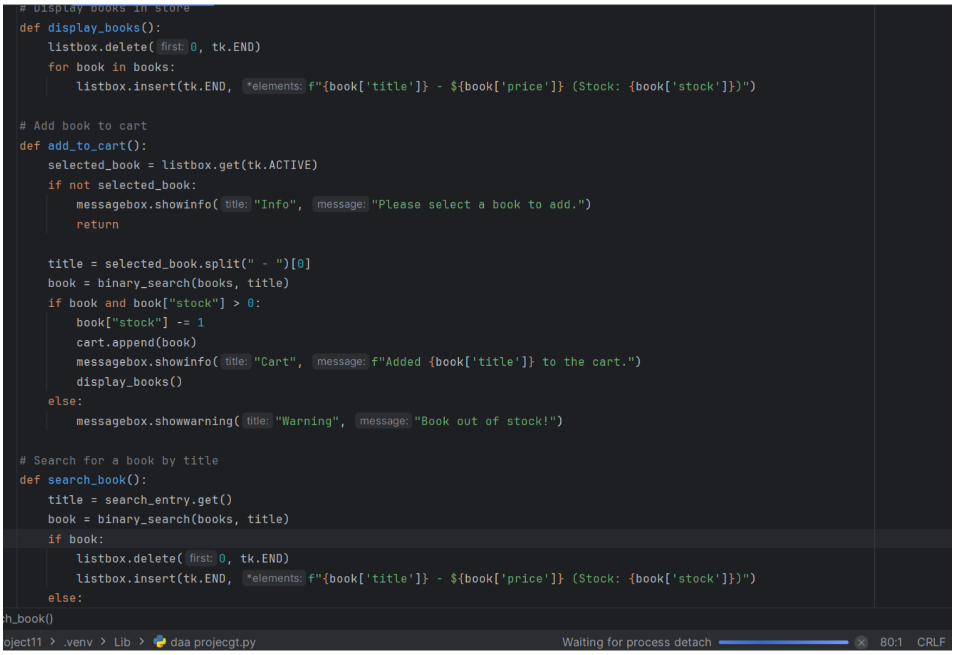
## SCREENSHOTS



```
import tkinter as tk
from tkinter import messagebox

# Book data
books = [
    {"title": "sense and sensability", "author": "jane austen", "price": 10.99, "stock": 5},
    {"title": "1984", "author": "George Orwell", "price": 8.99, "stock": 3},
    {"title": "it ends with us", "author": "coolen hoover", "price": 12.99, "stock": 4},
    {"title": "Pride and Prejudice", "author": "Jane Austen", "price": 7.99, "stock": 6},
]

# Sorting books alphabetically by title using merge sort
def merge_sort_books(books):
    if len(books) <= 1:
        return books
    mid = len(books) // 2
    left_half = merge_sort_books(books[:mid])
    right_half = merge_sort_books(books[mid:])
    return merge(left_half, right_half)

def merge(left, right):
    sorted_books = []
    while left and right:
        if left[0]["title"].lower() <= right[0]["title"].lower():
            sorted_books.append(left.pop(0))
        else:
            sorted_books.append(right.pop(0))
    sorted_books.extend(left or right)
    return sorted_books

books = merge_sort_books(books)
```

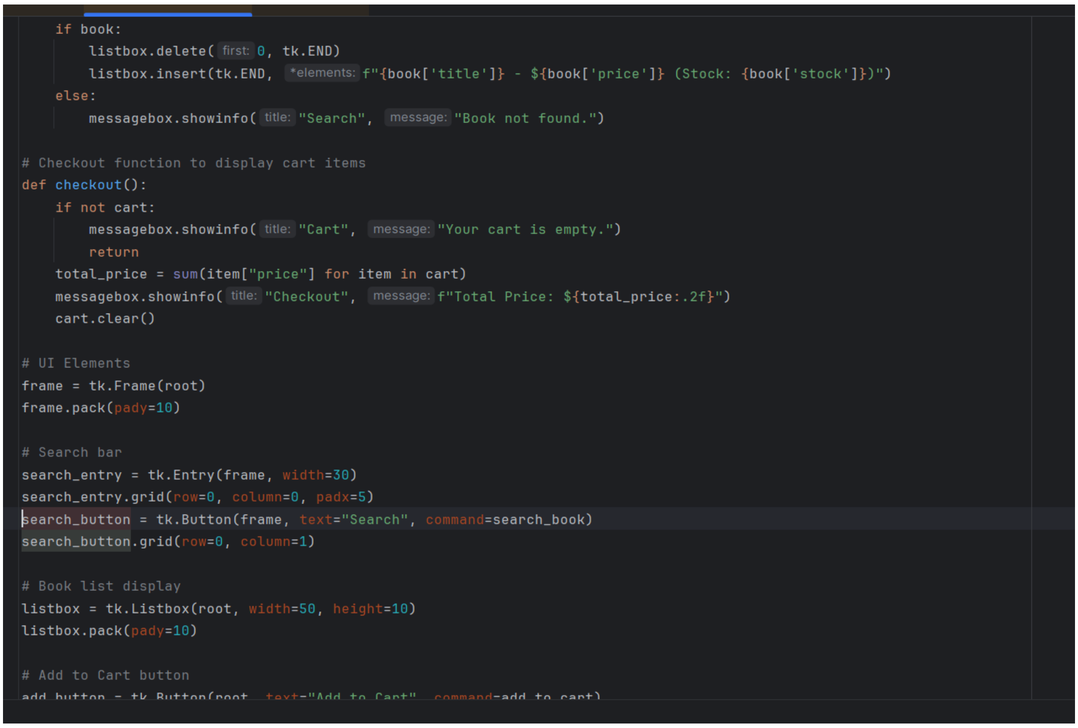


```
books = merge_sort_books(books)

# Search function (binary search)
def binary_search(books, title):
    low, high = 0, len(books) - 1
    while low <= high:
        mid = (low + high) // 2
        if books[mid]["title"].lower() == title.lower():
            return books[mid]
        elif books[mid]["title"].lower() < title.lower():
            low = mid + 1
        else:
            high = mid - 1
    return None

# Initialize main Tkinter window
root = tk.Tk()
root.title("Online Bookstore")

# Cart to store selected books
cart = []

# Display books in store
def display_books():
    listbox.delete( first: 0, tk.END)
    for book in books:
        listbox.insert(tk.END,  *elements: f"{book['title']} - ${book['price']} (Stock: {book['stock']})")

# Add book to cart
def add_to_cart():
```

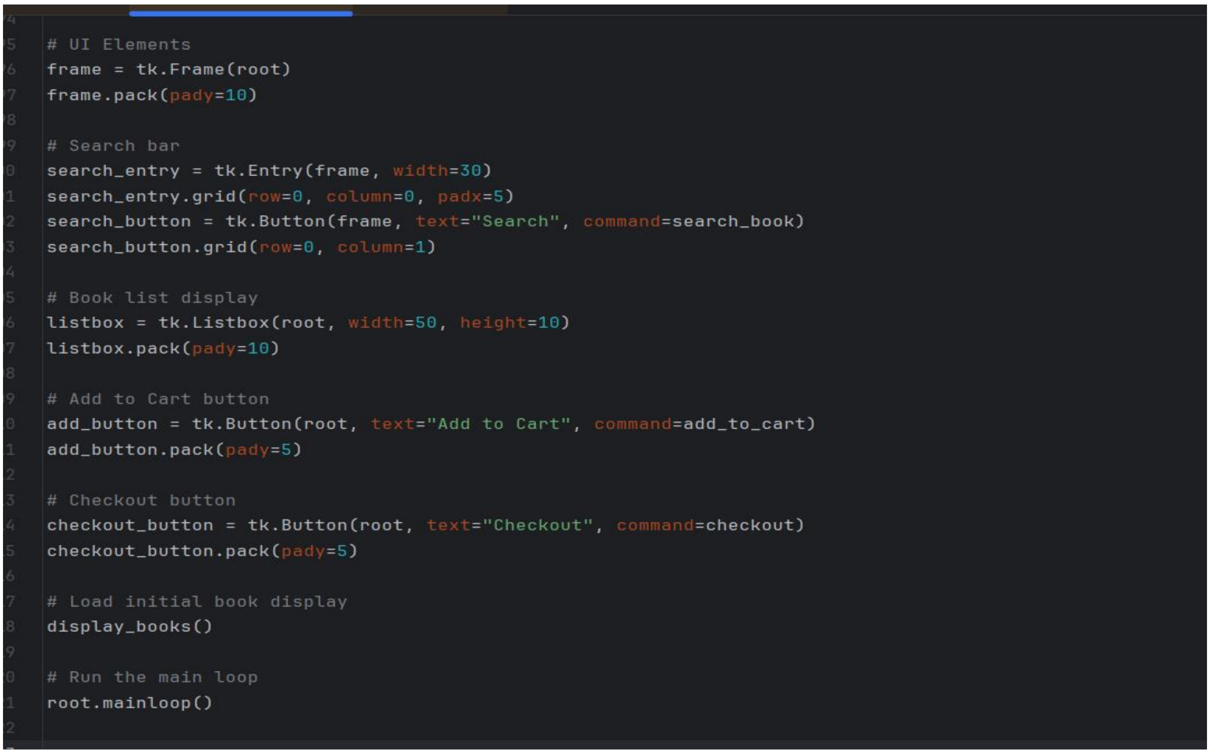


```
# Display books in store
def display_books():
    listbox.delete( first: 0, tk.END)
    for book in books:
        listbox.insert(tk.END,  *elements: f"{book['title']} - ${book['price']} (Stock: {book['stock']})")

# Add book to cart
def add_to_cart():
    selected_book = listbox.get(tk.ACTIVE)
    if not selected_book:
        messagebox.showinfo( title: "Info",  message: "Please select a book to add.")
        return

    title = selected_book.split(" - ")[0]
    book = binary_search(books, title)
    if book and book["stock"] > 0:
        book["stock"] -= 1
        cart.append(book)
        messagebox.showinfo( title: "Cart",  message: f"Added {book['title']} to the cart.")
        display_books()
    else:
        messagebox.showwarning( title: "Warning",  message: "Book out of stock!")

# Search for a book by title
def search_book():
    title = search_entry.get()
    book = binary_search(books, title)
    if book:
        listbox.delete( first: 0, tk.END)
        listbox.insert(tk.END,  *elements: f"{book['title']} - ${book['price']} (Stock: {book['stock']})")
    else:
```

h_book()
oject11 > .venv > Lib > daa projecgt.py
Waiting for process detach
80:1
CRLF

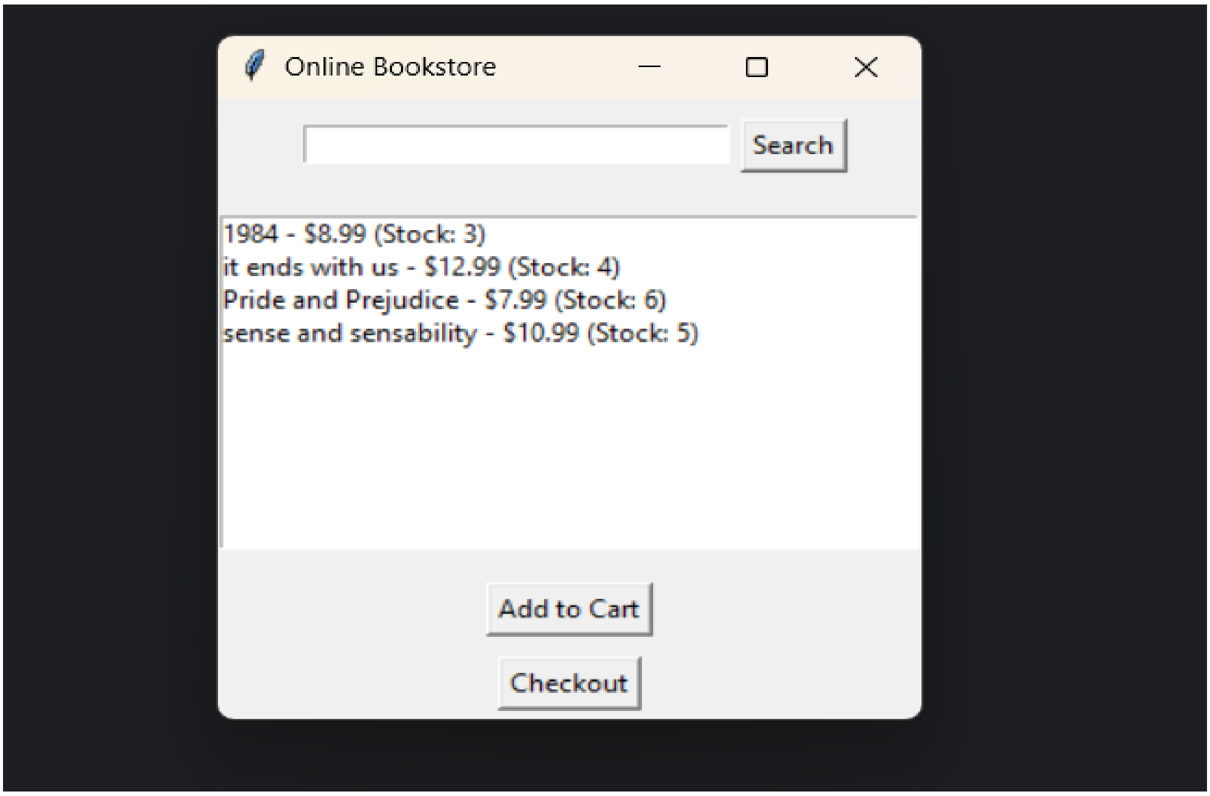


```
    if book:
        listbox.delete( first: 0, tk.END)
        listbox.insert(tk.END,  *elements: f"{book['title']} - ${book['price']} (Stock: {book['stock']})")
    else:
        messagebox.showinfo( title: "Search",  message: "Book not found.")

# Checkout function to display cart items
def checkout():
    if not cart:
        messagebox.showinfo( title: "Cart",  message: "Your cart is empty.")
        return
    total_price = sum(item["price"] for item in cart)
    messagebox.showinfo( title: "Checkout",  message: f"Total Price: ${total_price:.2f}")
    cart.clear()

# UI Elements
frame = tk.Frame(root)
frame.pack(pady=10)

# Search bar
search_entry = tk.Entry(frame, width=30)
search_entry.grid(row=0, column=0, padx=5)
search_button = tk.Button(frame, text="Search", command=search_book)
search_button.grid(row=0, column=1)

# Book list display
listbox = tk.Listbox(root, width=50, height=10)
listbox.pack(pady=10)

# Add to Cart button
add_button = tk.Button(root, text="Add to Cart", command=add_to_cart)
```



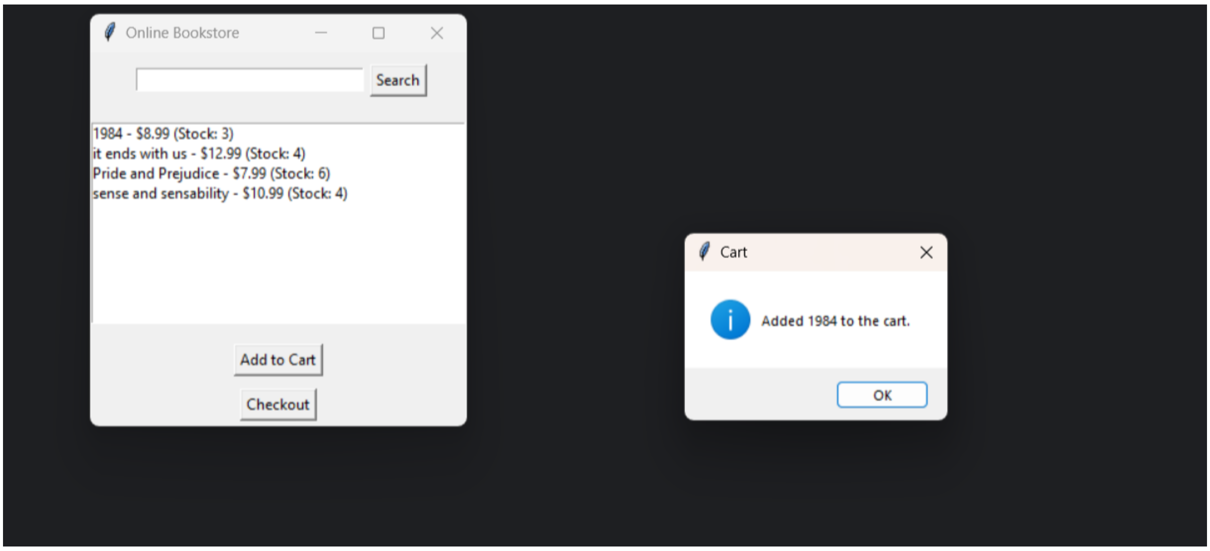
```
# UI Elements
frame = tk.Frame(root)
frame.pack(pady=10)

# Search bar
search_entry = tk.Entry(frame, width=30)
search_entry.grid(row=0, column=0, padx=5)
search_button = tk.Button(frame, text="Search", command=search_book)
search_button.grid(row=0, column=1)

# Book list display
listbox = tk.Listbox(root, width=50, height=10)
listbox.pack(pady=10)

# Add to Cart button
add_button = tk.Button(root, text="Add to Cart", command=add_to_cart)
add_button.pack(pady=5)

# Checkout button
checkout_button = tk.Button(root, text="Checkout", command=checkout)
checkout_button.pack(pady=5)

# Load initial book display
display_books()

# Run the main loop
root.mainloop()
```

## OUTPUT



1984 - $8.99 (Stock: 3)
it ends with us - $12.99 (Stock: 4)
Pride and Prejudice - $7.99 (Stock: 6)
sense and sensability - $10.99 (Stock: 5)

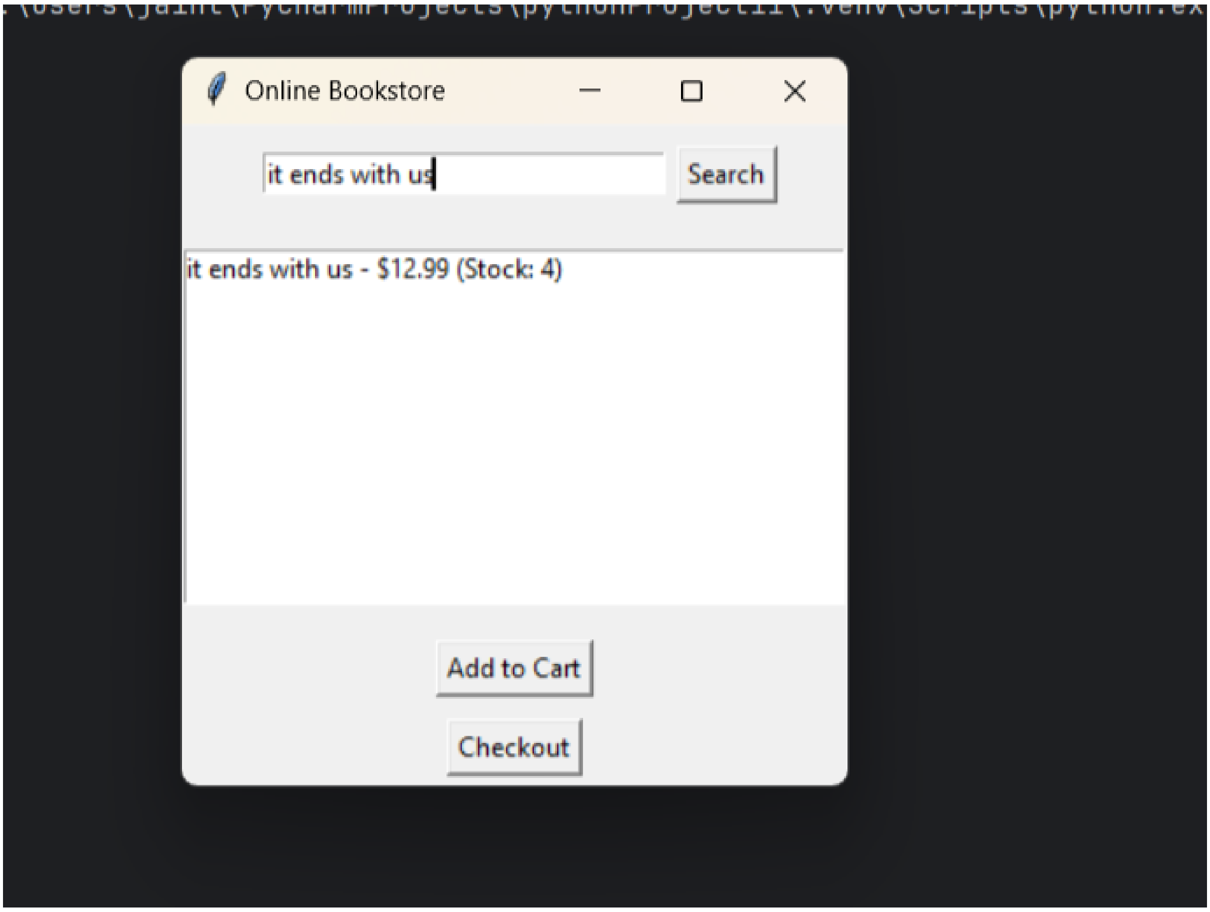
## ADD THE BOOK IN CART



1984 - $8.99 (Stock: 3)
it ends with us - $12.99 (Stock: 4)
Pride and Prejudice - $7.99 (Stock: 6)
sense and sensability - $10.99 (Stock: 4)

Added 1984 to the cart.

## Search for the book

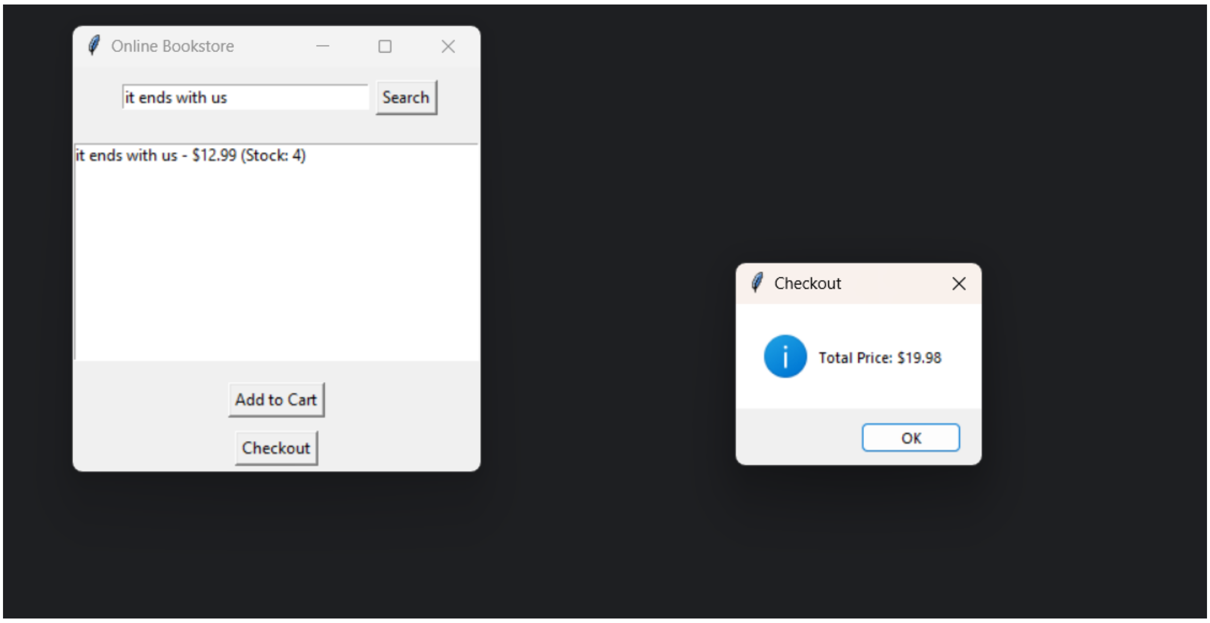


```
:\Users\jaint\PycharmProjects\pythonProject11\.venv\Scripts\python.ex
```

it ends with us

it ends with us - $12.99 (Stock: 4)

## Checkout



it ends with us

it ends with us - $12.99 (Stock: 4)

Total Price: $19.98

## Learning Outcomes

- Algorithm design and efficiency.
- Graphical user interface development.
- Data management and organization.
- Project development and problem solving.