class Item:

def \_\_init\_\_(self, title, author, price, stock=0):

self.title = title

self.author = author

self.price = price

self.stock = stock

def get\_details(self):

return f"Title: {self.title}, Author: {self.author}, Price: ${self.price}, Stock: {self.stock}"

def update\_details(self, \*\*kwargs):

for key, value in kwargs.items():

setattr(self, key, value)

def apply\_discount(self, discount\_percentage):

self.price -= (self.price \* (discount\_percentage / 100))

class Book(Item):

def \_\_init\_\_(self, title, author, price, isbn, genre, num\_pages, stock=0):

super().\_\_init\_\_(title, author, price, stock)

self.isbn = isbn

self.genre = genre

self.num\_pages = num\_pages

class Magazine(Item):

def \_\_init\_\_(self, title, author, price, issue\_number, publication\_date, editor, stock=0):

super().\_\_init\_\_(title, author, price, stock)

self.issue\_number = issue\_number

self.publication\_date = publication\_date

self.editor = editor

class DVD(Item):

def \_\_init\_\_(self, title, author, price, director, duration, genre, stock=0):

super().\_\_init\_\_(title, author, price, stock)

self.director = director

self.duration = duration

self.genre = genre

from datetime import datetime

class Customer:

def \_\_init\_\_(self, name, email, username, password):

self.name = name

self.email = email

self.username = username

self.password = password

self.cart = []

self.order\_history = []

def add\_to\_cart(self, item, quantity=1):

self.cart.append((item, quantity))

def remove\_from\_cart(self, item):

self.cart = [(i, q) for i, q in self.cart if i != item]

def view\_cart(self):

total\_price = 0

if not self.cart:

print("Your cart is empty.")

else:

print("Your cart:")

for item, quantity in self.cart:

total\_item\_price = item.price \* quantity

total\_price += total\_item\_price

print(f"{item.title} - Quantity: {quantity}, Price per item: ${item.price}, Total price: ${total\_item\_price}")

print(f"Total Price for all items: ${total\_price}")

def checkout(self):

total\_price = sum(item.price \* quantity for item, quantity in self.cart)

print(f"Total price for your items: ${total\_price}")

confirm = input("Do you want to proceed with the checkout? (yes/no): ")

if confirm.lower() == "yes":

print("Thank you for your purchase!")

self.order\_history.append((datetime.now(), self.cart))

self.cart = []

else:

print("Checkout canceled.")

def add\_item\_to\_cart(self, bookstore):

print("Available Items in Inventory:")

bookstore.view\_inventory()

item\_title = input("Enter the title of the item you want to add to cart: ")

quantity = int(input("Enter the quantity: "))

for item in bookstore.inventory:

if item.title.lower() == item\_title.lower():

if item.stock >= quantity:

self.add\_to\_cart(item, quantity)

item.stock -= quantity

print(f"{quantity} {item.title}(s) added to your cart.")

else:

print("Insufficient stock.")

return

print("Item not found in inventory.")

def remove\_item\_from\_cart(self, title):

for i, (item, quantity) in enumerate(self.cart):

if item.title.lower() == title.lower():

del self.cart[i]

print(f"{item.title} removed from your cart.")

return

print("Item not found in your cart.")

def view\_order\_history(self):

if not self.order\_history:

print("You have no previous orders.")

else:

print("Previous Orders:")

for order\_time, order\_items in self.order\_history:

print(f"Order placed on {order\_time}:")

for item, quantity in order\_items:

print(f"{item.title} - Quantity: {quantity}")

def calculate\_order\_total(self):

return sum(item.price \* quantity for item, quantity in self.cart)

class Staff:

def \_\_init\_\_(self, name, email, role):

self.name = name

self.email = email

self.role = role

class Bookstore:

def \_\_init\_\_(self):

self.inventory = []

self.customers = []

self.staff = [{"username": "OmarSaber", "password": "omar.saber", "role": "admin"}]

def login\_staff(self):

username = input("Enter username: ")

password = input("Enter password: ")

for staff\_member in self.staff:

if staff\_member["username"] == username and staff\_member["password"] == password:

return staff\_member

print("Invalid username or password.")

return None

def add\_item(self, item, staff):

if staff["role"] == "admin":

self.inventory.append(item)

print(f"{item.title} added to inventory.")

else:

print("You don't have permission to perform this action.")

def remove\_item(self, title, staff):

if staff["role"] == "admin":

for item in self.inventory:

if item.title.lower() == title.lower():

self.inventory.remove(item)

print(f"{item.title} removed from inventory.")

break

else:

print("Item not found in inventory.")

else:

print("You don't have permission to perform this action.")

def view\_inventory(self, staff=None):

if staff is None or staff["role"] != "admin":

print("Available options for customers:")

print("5. Search Items by Title")

print("6. Search Items by Author")

print("7. Search Items by Genre")

print("8. View Cart")

print("9. Checkout")

else:

print("Inventory:")

for item in self.inventory:

print(item.get\_details())

def search\_items\_by\_title(self, title):

matching\_items = [item for item in self.inventory if title.lower() in item.title.lower()]

if matching\_items:

print("Matching Items:")

for item in matching\_items:

print(item.get\_details())

else:

print("No items found with that title.")

def search\_items\_by\_author(self, author):

matching\_items = [item for item in self.inventory if author.lower() in item.author.lower()]

if matching\_items:

print("Matching Items:")

for item in matching\_items:

print(item.get\_details())

else:

print("No items found by that author.")

def search\_items\_by\_genre(self, genre):

matching\_items = [item for item in self.inventory if genre.lower() in item.genre.lower()]

if matching\_items:

print("Matching Items:")

for item in matching\_items:

print(item.get\_details())

else:

print("No items found in that genre.")

def add\_customer(self, name, email, username, password):

customer = Customer(name, email, username, password)

self.customers.append(customer)

print(f"Customer {name} added.")

def view\_customers(self):

print("Customers:")

for customer in self.customers:

print(customer.name)

def login\_customer(self):

username = input("Enter username: ")

password = input("Enter password: ")

for customer in self.customers:

if customer.username == username and customer.password == password:

return customer

print("Invalid username or password.")

return None

bookstore = Bookstore()

while True:

print("\nAre you a staff member? (yes/no)")

is\_staff = input().lower()

if is\_staff == "yes":

user = bookstore.login\_staff()

if user is None:

continue

elif is\_staff == "no":

user = None

else:

print("Invalid input.")

continue

if user is not None and user["role"] == "admin":

print("\n1. Add Book")

print("2. Add Magazine")

print("3. Add DVD")

print("4. View Inventory")

print("10. Add Customer")

print("11. View Customers")

choice = input("Enter your choice: ")

if choice == "1" or choice == "2" or choice == "3":

if user is not None and user["role"] == "admin":

if choice == "1":

title = input("Enter book title: ")

author = input("Enter author name: ")

price = float(input("Enter price: "))

isbn = input("Enter ISBN: ")

genre = input("Enter genre: ")

num\_pages = int(input("Enter number of pages: "))

stock = int(input("Enter stock: "))

book = Book(title, author, price, isbn, genre, num\_pages, stock)

bookstore.add\_item(book, user)

elif choice == "2":

title = input("Enter magazine title: ")

author = input("Enter author name: ")

price = float(input("Enter price: "))

issue\_number = int(input("Enter issue number: "))

publication\_date = input("Enter publication date: ")

editor = input("Enter editor: ")

stock = int(input("Enter stock: "))

magazine = Magazine(title, author, price, issue\_number, publication\_date, editor, stock)

bookstore.add\_item(magazine, user)

else:

title = input("Enter DVD title: ")

author = input("Enter author name: ")

price = float(input("Enter price: "))

director = input("Enter director: ")

duration = input("Enter duration: ")

genre = input("Enter genre: ")

stock = int(input("Enter stock: "))

dvd = DVD(title, author, price, director, duration, genre, stock)

bookstore.add\_item(dvd, user)

else:

print("You don't have permission to perform this action.")

elif choice == "4":

bookstore.view\_inventory(user)

elif choice == "10":

name = input("Enter customer name: ")

email = input("Enter customer email: ")

username = input("Enter customer username: ")

password = input("Enter customer password: ")

bookstore.add\_customer(name, email, username, password)

elif choice == "11":

bookstore.view\_customers()

else:

print("Invalid choice.")

else:

print("\n1. Login as Customer")

choice = input("Enter your choice: ")

if choice == "1":

customer = bookstore.login\_customer()

if customer:

while True:

print("\n1. Search Items by Title")

print("2. Search Items by Author")

print("3. Search Items by Genre")

print("4. View Cart")

print("5. Add Item to Cart")

print("6. Checkout")

print("7. Logout")

print("8. View Order History")

print("9. Remove from cart")

customer\_choice = input("Enter your choice: ")

if customer\_choice == "1":

title = input("Enter the title to search: ")

bookstore.search\_items\_by\_title(title)

elif customer\_choice == "2":

author = input("Enter the author to search: ")

bookstore.search\_items\_by\_author(author)

elif customer\_choice == "3":

genre = input("Enter the genre to search: ")

bookstore.search\_items\_by\_genre(genre)

elif customer\_choice == "4":

customer.view\_cart()

elif customer\_choice == "5":

customer.add\_item\_to\_cart(bookstore)

elif customer\_choice == "6":

customer.checkout()

elif customer\_choice == "7":

print("Logging out.")

break

elif customer\_choice == "8":

customer.view\_order\_history()

elif customer\_choice == "9":

title\_to\_remove = input("Enter the title of the item to remove from your cart: ")

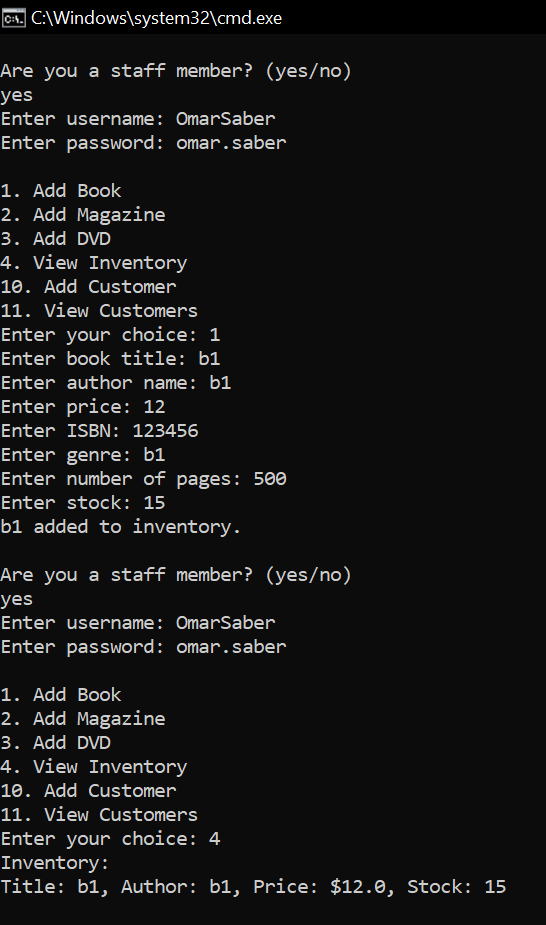
customer.remove\_item\_from\_cart(title\_to\_remove)

else:

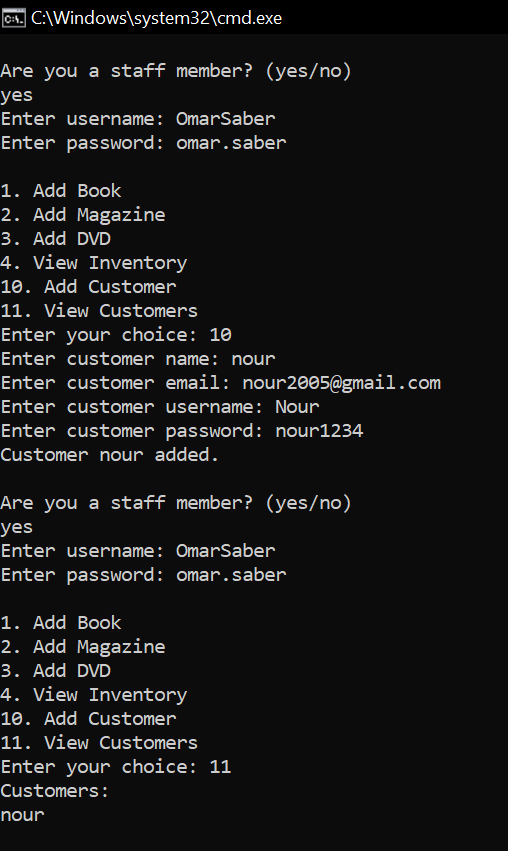
print("Invalid choice.")

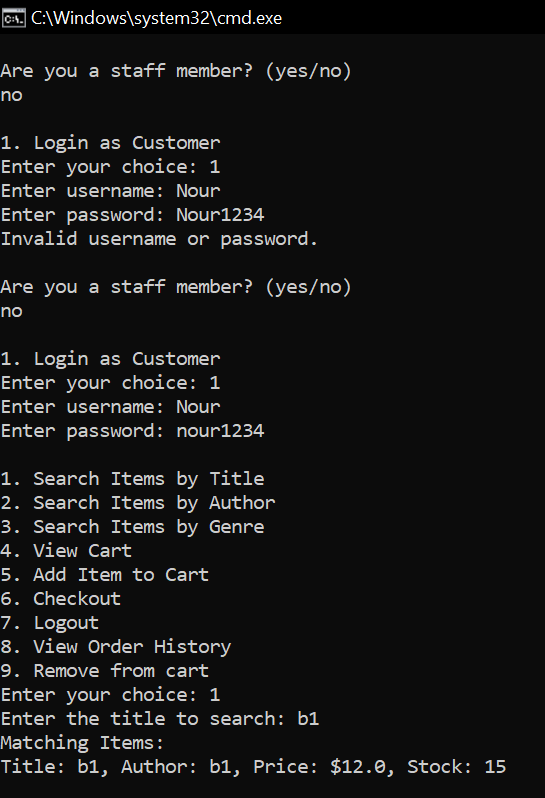
else:

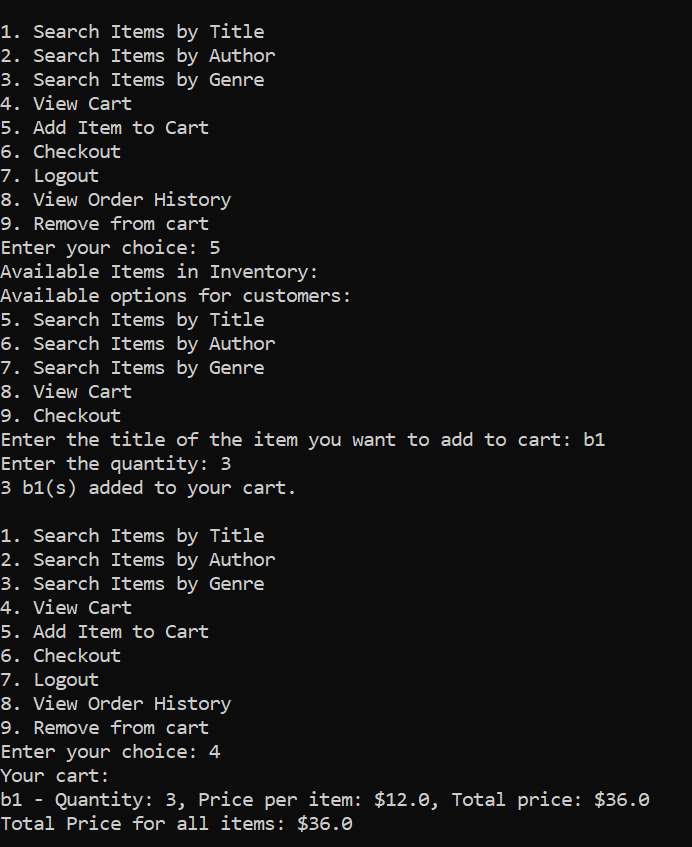
print("Invalid choice.")

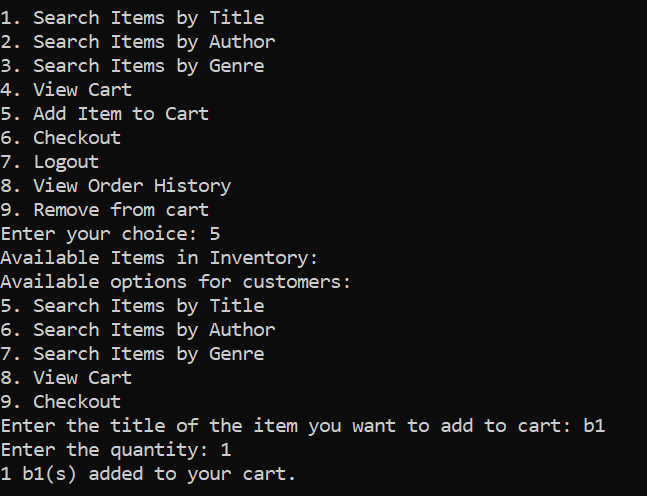


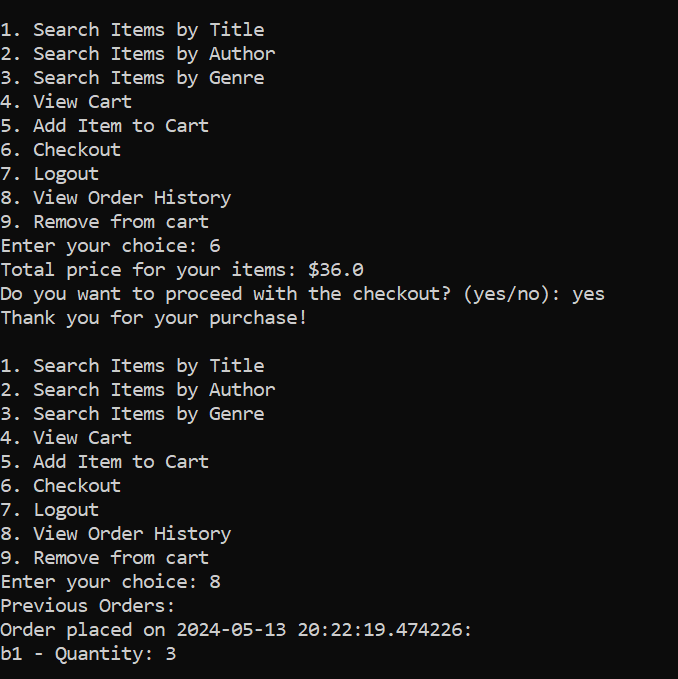
Adding book and viewing inventory

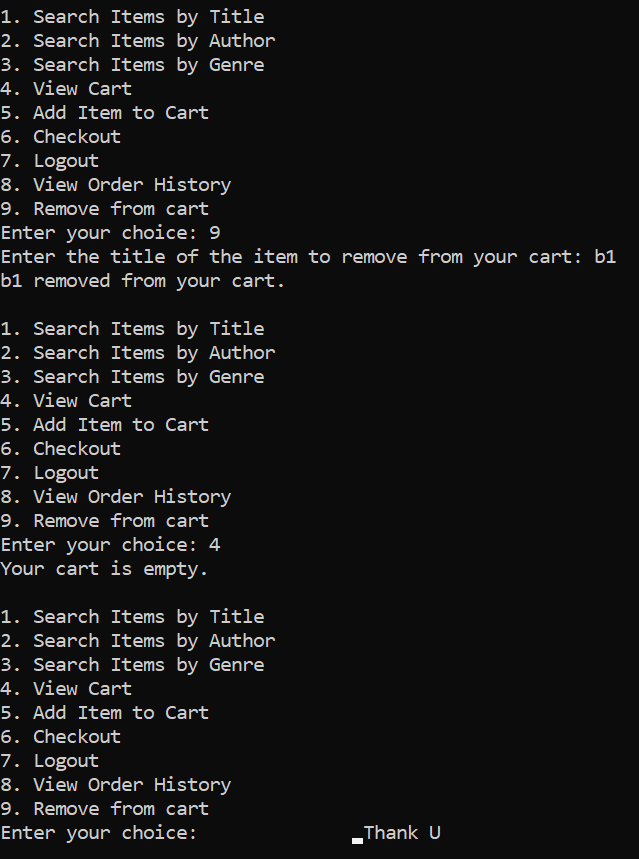
adding customer and viewing customers

logging in as customer ,searching without adding to cart

adding to cart , viewing cart

adding to cart

checking out , viewing customer history



Removing from cart , viewing cart after removing