# Portfolio Milestone 2

**Steps**:

* Define ShoppingCart Class with attributes customer name, current date, cart items list(Zybooks,2019).
* Define parameterized constructor with default constructor(GeeksforGeek,2023).

class ShoppingCart:

customer\_name : str

current\_date : str

cart\_items = []

# Default Constructor to initialize customer name and date

def \_\_init\_\_(self, name='none', date = 'January 1,2020'):

self.customer\_name = name

self.current\_date = date

* Create Method for add items, Modify items, Remove items.
  + Add Items Method : Has parameter - itemToPurchase

Append item to Cart Item List (GeeksforGeek,2023).

def add\_item(self,temp\_item):

self.cart\_items.append(temp\_item)

* + Modify Item Method: Modifies an item's description, price, and/or quantity.

Has parameter - ItemToPurchase. If an item is present in the cart, the parameter doesn’t have a default value. modify the item in the cart. If an item is not present in the cart, it shows a message - Item not found in cart (Zybooks,2019).

**Logic** - Since it is a List object, the program uses index to perform action on the cart items list. Here, ‘for loop’ will run for a range from 0 to length of cart\_items List.

The program compares if specific values from user input is default value and only updates the record if it’s not default.

It sets flag item\_found to true and breaks the loop (as we already found the value, no need to run the loop again. **Assumption** - only one unique record with the same item name.) If an item is not found in the cart, it shows output: Item not found in cart.

def modify\_item(self,temp\_item):

for i in range(len(self.cart\_items)):

if self.cart\_items[i].item\_name == temp\_item.item\_name:

if( int(temp\_item.item\_price) != 0):

self.cart\_items[i].item\_price = temp\_item.item\_price

if( temp\_item.item\_quantity != 0):

self.cart\_items[i].item\_quantity = temp\_item.item\_quantity

if( temp\_item.description != ''):

self.cart\_items[i].description = temp\_item.description

#shopping\_cart.cart\_items.update(temp\_item)

item\_found = True

break

if(not item\_found):

print('Item not found in cart.')

* + Remove item Method : Has Parameter - String: item\_name

It runs the loop for cart\_items list, searches the item with attribute item\_name as parameter. Remove the indexed element using the del method (GeekforGeeks,2023). If no records are found, it shows a message that Item is not found.

def remove\_item(self,rmv\_item):

for i in range(len(self.cart\_items)):

if self.cart\_items[i].item\_name == rmv\_item:

del self.cart\_items[i]

item\_found= True

break

if(not item\_found):

print('Item not found in cart.')

* + Method - get\_num\_items\_in\_cart : Doesn’t accept any parameter.

It loops through all cart\_items and adds quantities of items within the list and returns the count.

def get\_num\_items\_in\_cart(self):

total\_count = 0

for item in self.cart\_items:

total\_count += item.item\_quantity

return total\_count

An Alternative method to calculate total is using the sum function.

total\_count = sum(i.quantity for i in self.cart\_items) (Built-in Types, n.d.-c)

* + Method - get\_cost\_of\_cart : Doesn’t accept any parameter.

Iterate through all cart\_items, calculate the total cost of items within the cart and return the Total Cost (Zybooks,2019).

for item in self.cart\_items:

Total\_Cost += (item.item\_price \* item.item\_quantity)

return Total\_Cost

* + Method - print\_total : Doesn’t accept any parameter. Calls get\_cost\_of\_cart to get total cost of cart items. Calls ItemtoPurchase’s print\_item\_cost to print each item. Format the output (Zybooks, 2019).

print(shopping\_cart.customer\_name,'\' Shopping Cart -', self.current\_date)

if self.cart\_items:

print('\tNumber of Items:', self.get\_num\_items\_in\_cart())

for item in shopping\_cart.cart\_items:

item.print\_item\_cost()

print('\n\tTotal:', '${:.2f}'.format(total\_cart\_Cost))

print('==============================')

print('Thank you for shopping with us!\n')

else:

print('SHOPPING CART IS EMPTY.')

* + Method - print\_item\_description: No parameters. If there are items in the cart, get the description of each item by iterating them (GeekforGeeks,2023).

if self.cart\_items:

print('\tItem Descriptions')

for item in self.cart\_items:

print(item.item\_name, ':', item.description)

else:

print('SHOPPING CART IS EMPTY.')

**Step: 5**

* **Print Menu**:

Define method print\_menu. Accepts parameter as shopping\_cart object. It prints menu options for users to view and select (Zybooks,2019).

def print\_menu(obj\_cart):

print('\t\tMENU\n')

print('\ta - Add item to cart')

print('\tr - Remove item from cart')

print('\tc - Change item quantity')

print('\ti - Output items \'descriptions\'')

print('\to - Output shopping cart')

print('\tq - Quit')

print('\t\tChoose an option:\n')

* **Main** -
  + Program creates shopping cart object (default/based on user input)
  + It runs Do\_while loop setting limit of Max\_ITEM to avoid infinite loop (Python, 2022).
  + It calls print\_menu() and prints Menu for User to insert the selected option.
  + Implement quit option to make sure, this while loop doesn’t go into infinite loop while testing locally. It will capture user input and as soon it receives ‘q’, it will break the loop and exit (Python, 2022).
  + Based on User’s input, it calls specific methods defined earlier. Here, for the add and modify option, it prompts the user to insert values and stores the data into ItemToPurchase object and passes that as a parameter to the appropriate method (Gaddis, 2017).
  + If a user inserts invalid options, the program will prompt the user to enter a valid option.
* MAX\_ITEM = 10
* item\_count = 0
* if(c\_name and c\_date):
* shopping\_cart = ShoppingCart(c\_name,c\_date)
* else:
* shopping\_cart = ShoppingCart()
* while (item\_count< MAX\_ITEM):
* print\_menu(shopping\_cart)
* user\_input = input().lower()
* if user\_input == 'a':
* temp\_item = ItemToPurchase()
* # assign user input value via attribute assignment
* try:
* temp\_item.item\_name = input('Enter Item Name: \n')
* temp\_item.description = input('Enter Description:\n')
* temp\_item.item\_price = float(input('Enter Item Price: \n'))
* temp\_item.item\_quantity = int(input('Enter Item Quantity: \n'))
* shopping\_cart.add\_item(temp\_item)
* item\_count += 1
* except:
* print('Invalid Input.')
* elif user\_input == 'r':
* rm\_item = input('Please Enter Item Name to Remove.')
* shopping\_cart.remove\_item(rm\_item)
* elif user\_input == 'c':
* temp\_item = ItemToPurchase()
* # assign user input value via attribute assignment
* try:
* temp\_item.item\_name = input('Enter Item Name to update: \n')
* temp\_item.description = input('Enter Description to update: \n')
* temp\_item.item\_price = float(input('Enter Item Price to update: \n'))
* temp\_item.item\_quantity = int(input('Enter Item Quantity to update: \n'))
* except:
* print('Invalid Input.')
* shopping\_cart.modify\_item(temp\_item)
* elif user\_input == 'i':
* print('\tOUTPUT ITEMS\' DESCRIPTIONS')
* shopping\_cart.print\_descriptions()
* elif user\_input == 'o':
* print('\tOUTPUT SHOPPING CART')
* shopping\_cart.print\_total()
* elif user\_input == 'q':
* print('You have selected to Exit.')
* print('Thank you for shopping with us!\n')
* break
* else:
* print('Please select valid input or press q to Quit.')

**Step 6:** Implement Output shopping cart menu option. Implement Output item's description menu option.

* **Output Shopping Cart** - The user selects option - ‘o’ from the menu by typing o or O.
  + The user input is converted to lower and compared for the conditions.

user\_input = input().lower()

* + Calls the method print\_total()

elif user\_input == 'o':

print('\tOUTPUT SHOPPING CART')

shopping\_cart.print\_total()

* + Print\_total calls get\_cost\_of\_cart method to get total Cart value.
  + If there are items in the list then, Program iterates over the list and call

print\_item\_cost() to print the individual item and the cost of each item.

* + Format the total\_cost via string formatter and print the value.
  + Prints User Friendly output format (Zybooks, 2019) (Section - Method - print\_total).
  + If the cart is empty then it displays a message that ‘SHOPPING CART IS EMPTY.’
* **Output item's description**: The user selects option - ‘i’ from the menu by typing i or I.
  + If there are items in the cart, get the description of each item by iterating them (GeekforGeeks,2023).

if self.cart\_items:

print('\tItem Descriptions')

for item in self.cart\_items:

print(item.item\_name, ':', item.description)

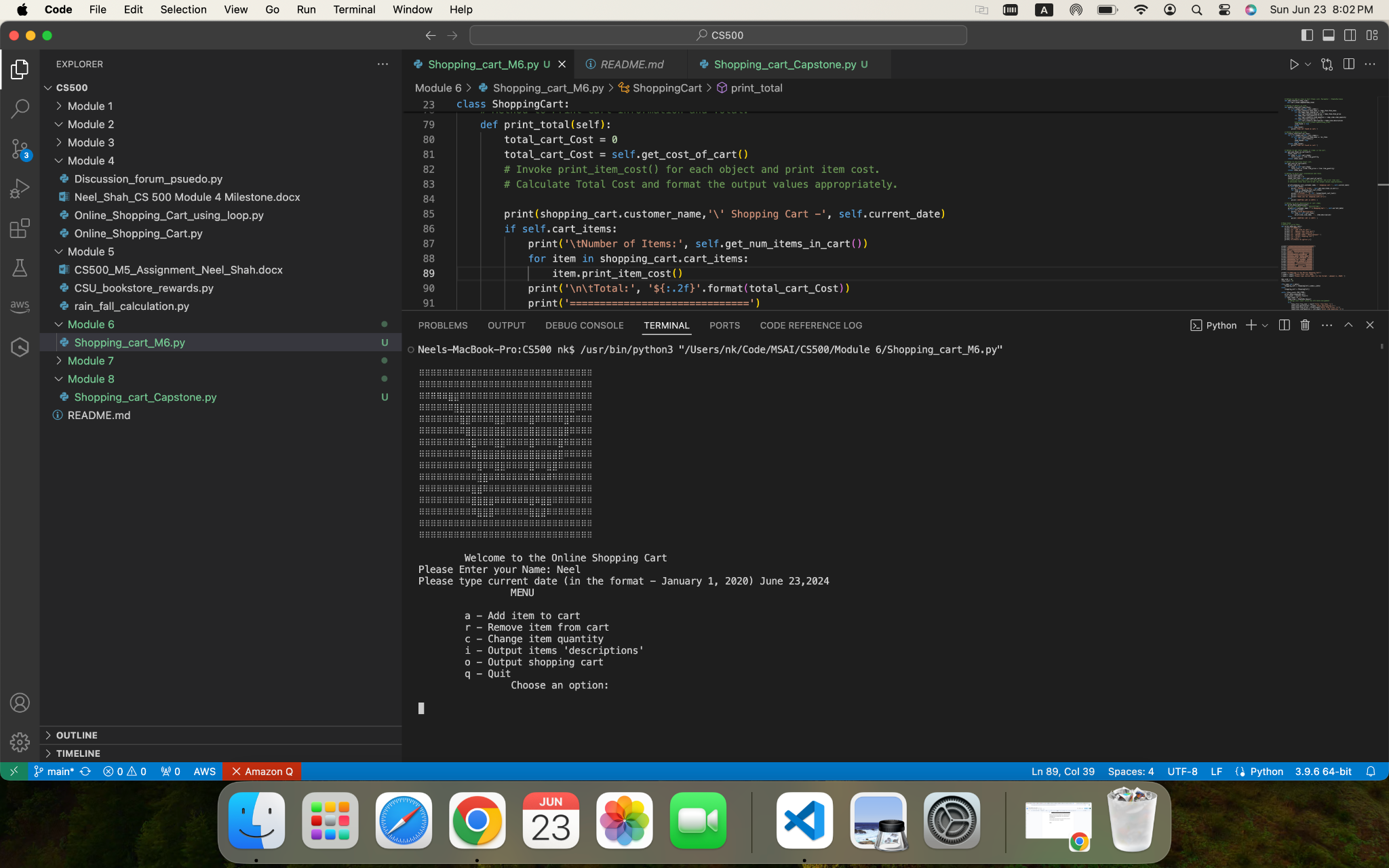
else:

print('SHOPPING CART IS EMPTY.')

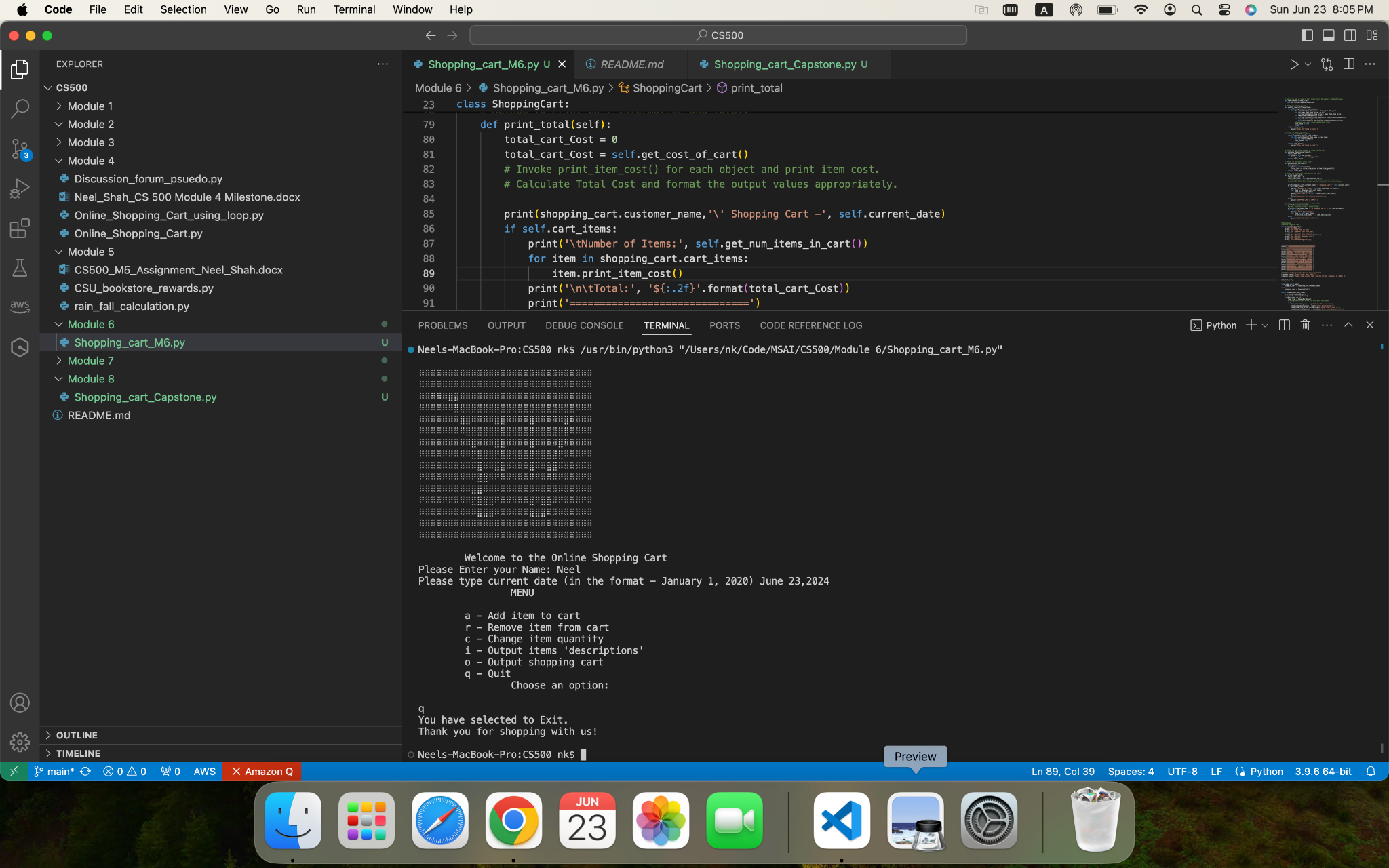
**Note**: Add, modify and remove are implemented here. I realized that the detailed implementation is part of the Capstone Project. I had not noticed it earlier, so I have included it here.

However, In reality, the add, modify and remove method would be a stub method which can display ‘Under Construction… Coming Soon.’

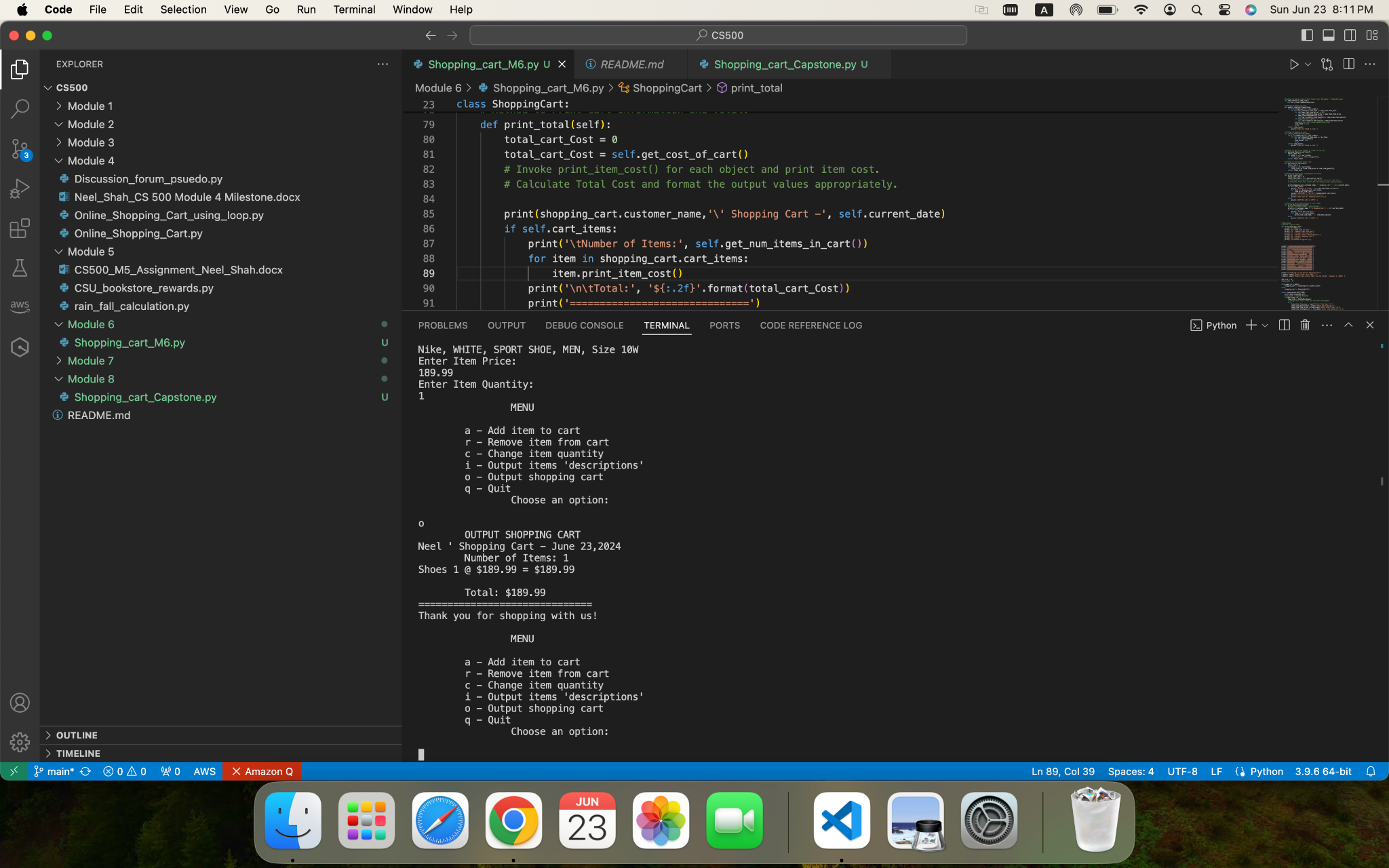
**Output**: Screenshot - Print Menu



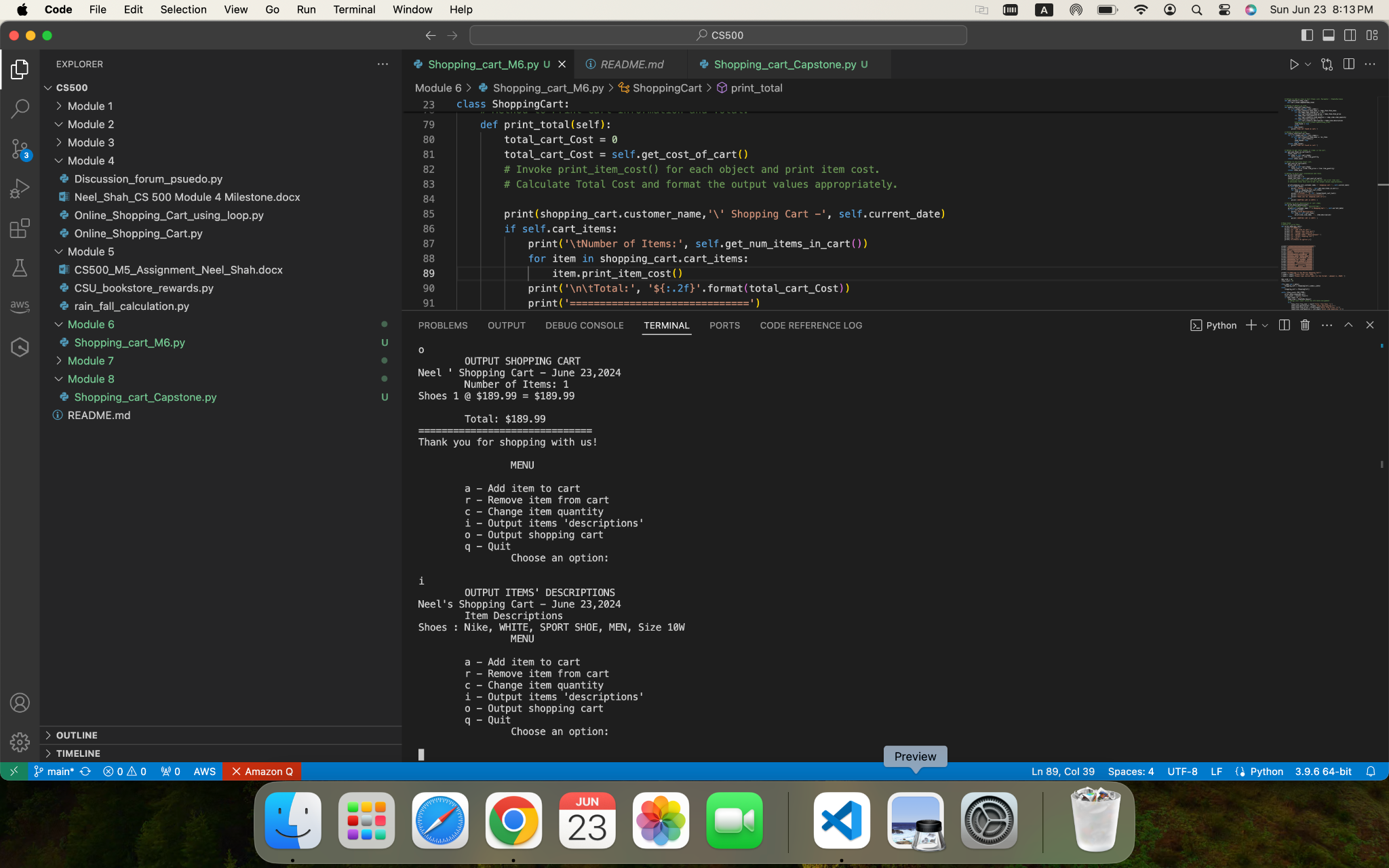
Screenshot - Selecting Quit Option



Screenshot - Selecting Output Shopping Cart Option



Screenshot - Output Items ‘descriptions’



**Source Code Path:**

<https://github.com/nshahcsu/MSAI/blob/main/CS500/Module%206/Shopping_cart_M6.py>

**References:**

Built-in types. (n.d.-c). Python Documentation. https://docs.python.org/3/library/stdtypes.html#lists

Gaddis, T. (2017). Starting Out with Python (4th Edition).

GeeksforGeeks. (2023, March 1). *Constructors in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/constructors-in-python/

GeeksforGeeks. (2024, April 5). Python Functions. GeeksforGeeks. <https://www.geeksforgeeks.org/python-functions/>

GeeksforGeeks. (2023b, April 7). Python Remove elements at Indices in List. GeeksforGeeks. https://www.geeksforgeeks.org/python-remove-elements-at-indices-in-list/#

GeeksforGeeks. (2023, April 13). Python try except. GeeksforGeeks. https://www.geeksforgeeks.org/python-try-except/

Python, R. (2022, July 18). Python “while” Loops (Indefinite Iteration). https://realpython.com/python-while-loop/

ZyBooks. (2019 August). CSC500: Principles of Programming. Module 2.19.