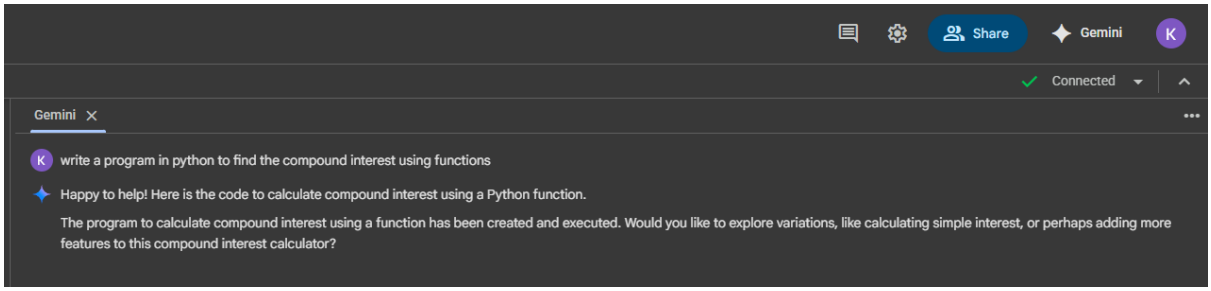
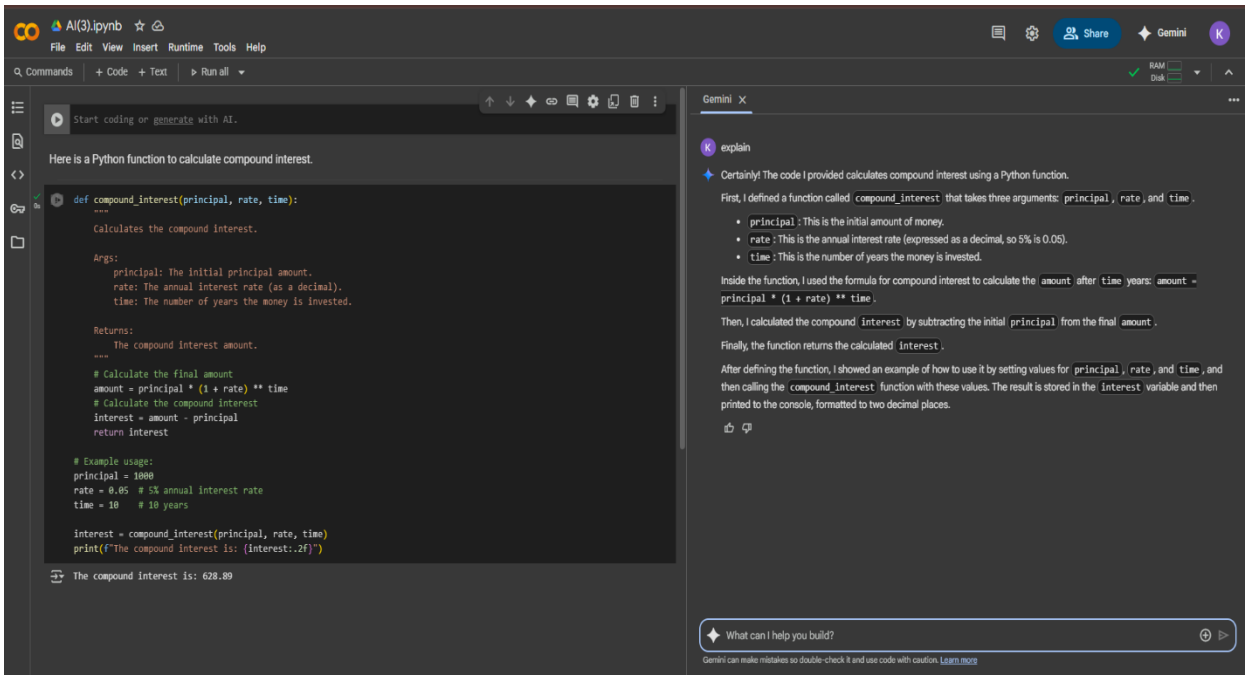
Assignment 3.2

Name:N.JYOSHNA SRI

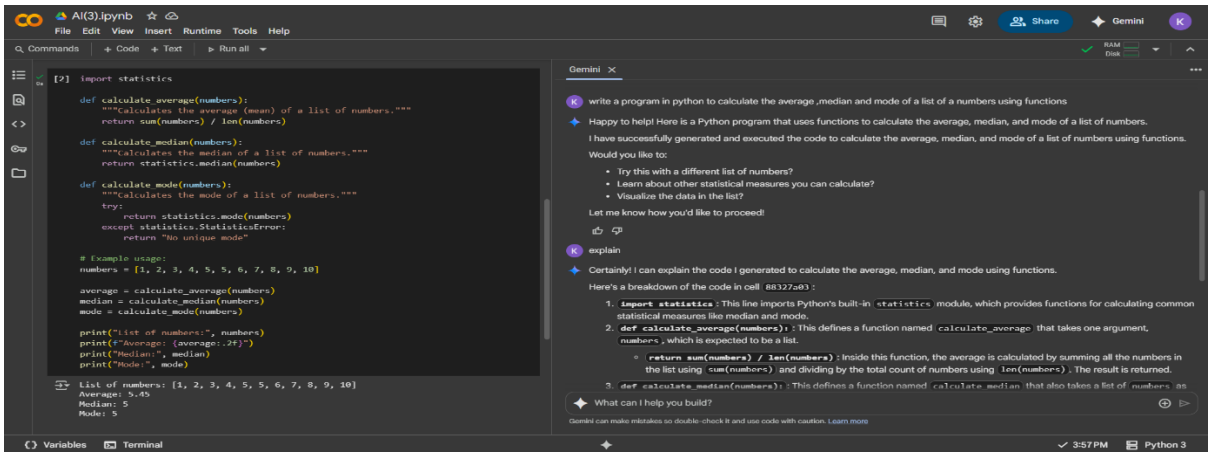
2403A52106

Task 1:

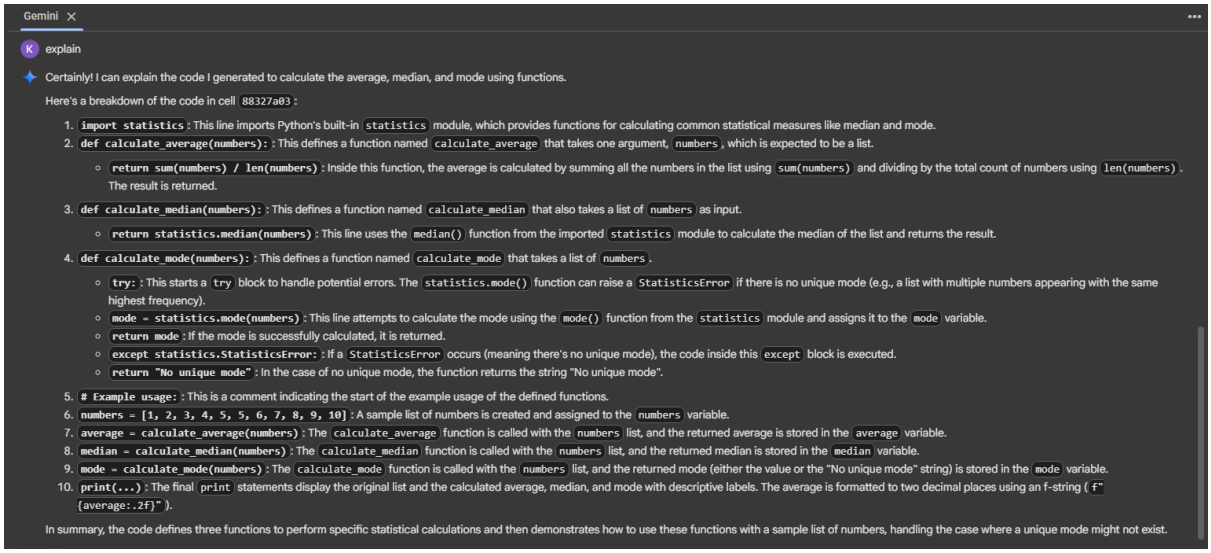




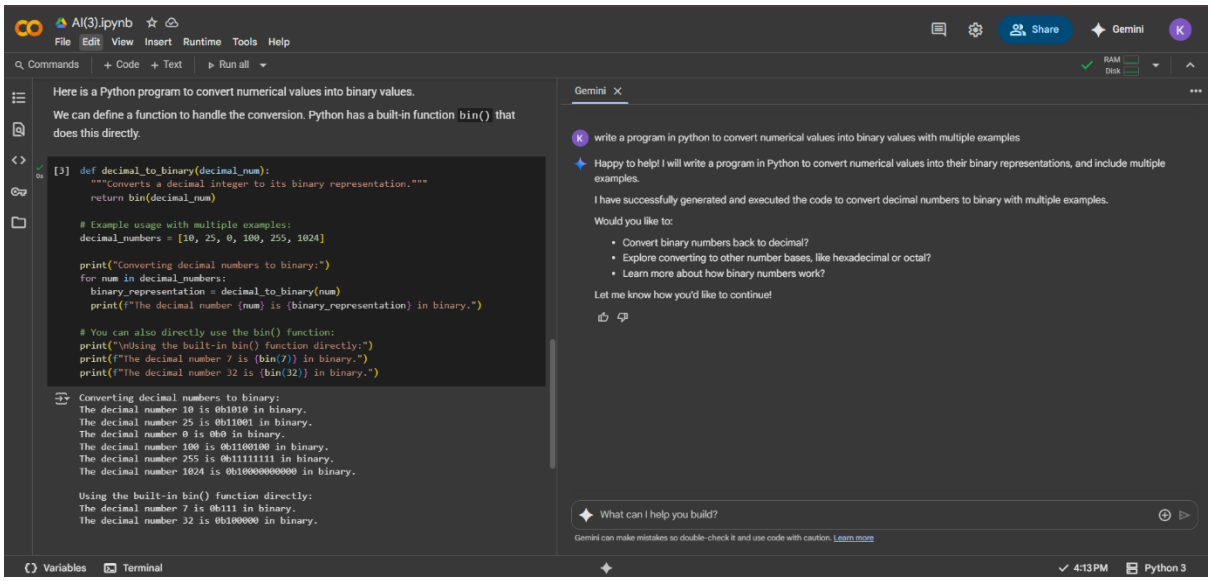
Task 2:



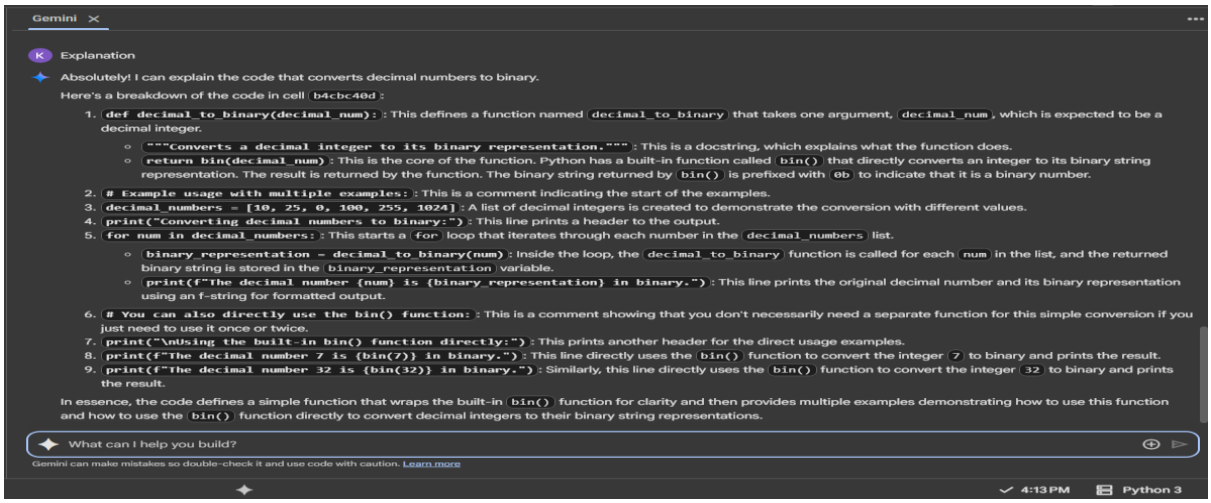
Explanation:



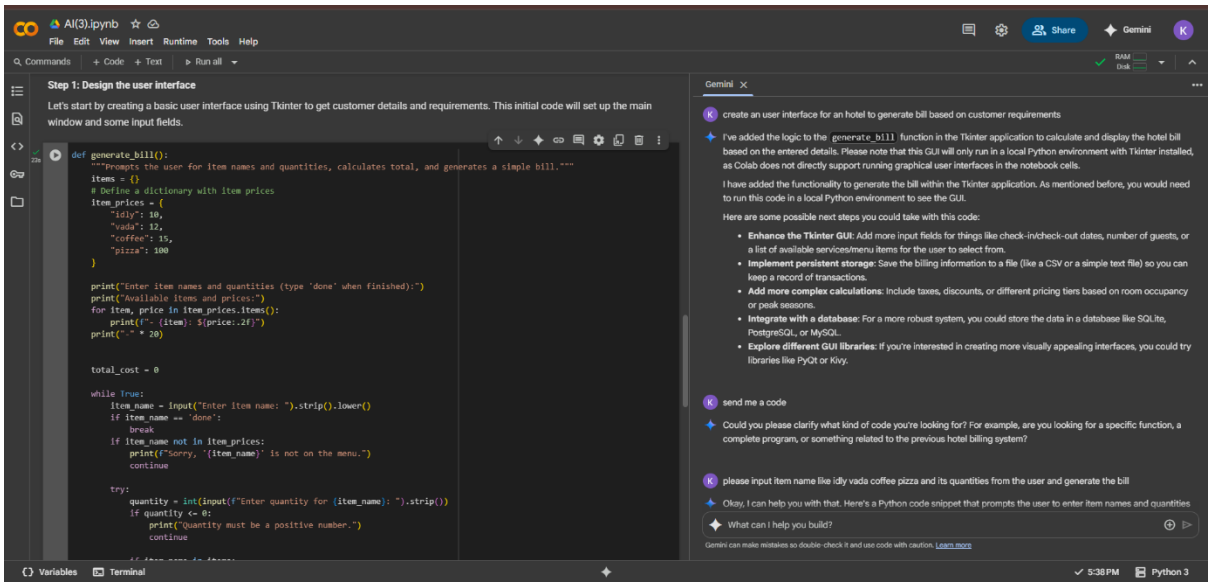
Task 3:

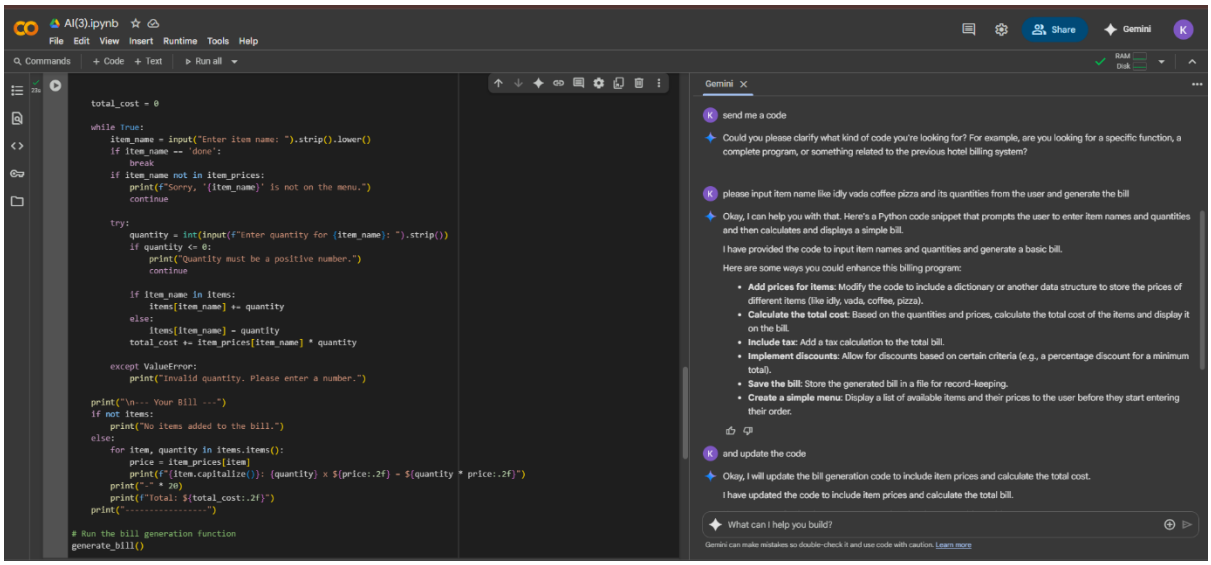


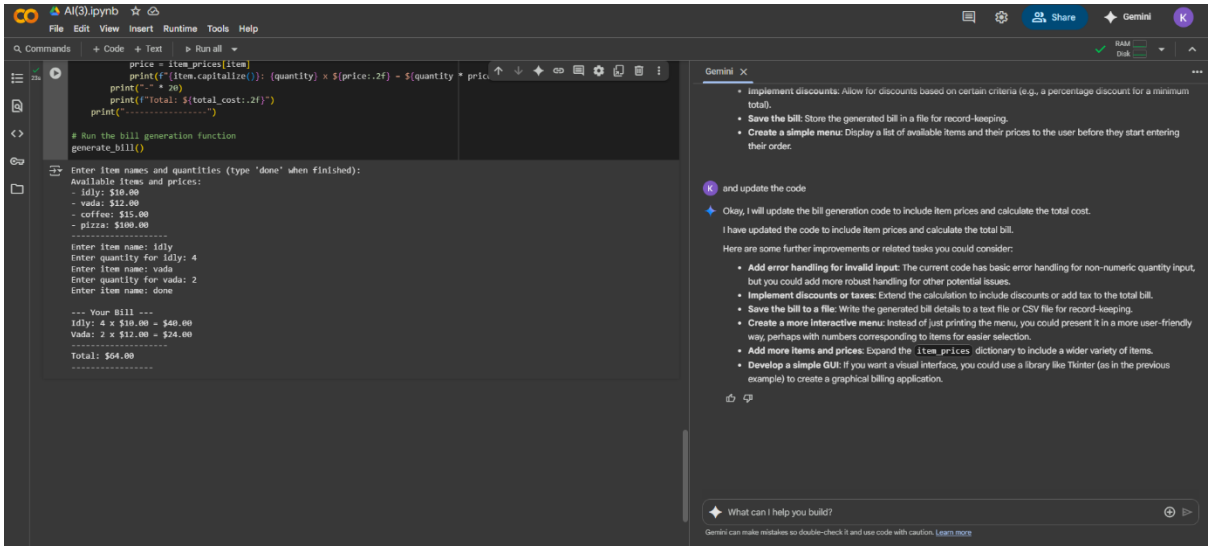
Explanation:



Task 4:


AI(3).ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands + Code + Text Run all
Step 1: Design the user interface
Let's start by creating a basic user interface using Tkinter to get customer details and requirements. This initial code will set up the main window and some input fields.
def generate_bill():
"""Prompts the user for item names and quantities, calculates total, and generates a simple bill."""
items = {}
# Define a dictionary with item prices
item_prices = {
"idly": 10,
"vada": 12,
"coffee": 15,
"pizza": 100
}
print("Enter item names and quantities (type 'done' when finished):")
print("Available items and prices:")
for item, price in item_prices.items():
print(f"- {item}: ${price:.2f}")
print("-" * 20)
total_cost = 0
while True:
item_name = input("Enter item name: ").strip().lower()
if item_name == 'done':
break
if item_name not in item_prices:
print(f"Sorry, '{item_name}' is not on the menu.")
continue
try:
quantity = int(input(f"Enter quantity for {item_name}: ").strip())
if quantity <= 0:
print("Quantity must be a positive number.")
continue
Gemini
create an user interface for an hotel to generate bill based on customer requirements
I've added the logic to the generate_bill function in the Tkinter application to calculate and display the hotel bill based on the entered details. Please note that this GUI will only run in a local Python environment with Tkinter installed, as Colab does not directly support running graphical user interfaces in the notebook cells.
I have added the functionality to generate the bill within the Tkinter application. As mentioned before, you would need to run this code in a local Python environment to see the GUI.
Here are some possible next steps you could take with this code:
Enhance the Tkinter GUI: Add more input fields for things like check-in/check-out dates, number of guests, or a list of available services/menu items for the user to select from.
Implement persistent storage: Save the billing information to a file (like a CSV or a simple text file) so you can keep a record of transactions.
Add more complex calculations: Include taxes, discounts, or different pricing tiers based on room occupancy or peak seasons.
Integrate with a database: For a more robust system, you could store the data in a database like SQLite, PostgreSQL, or MySQL.
Explore different GUI libraries: If you're interested in creating more visually appealing interfaces, you could try libraries like PyQt or Kivy.
send me a code
Could you please clarify what kind of code you're looking for? For example, are you looking for a specific function, a complete program, or something related to the previous hotel billing system?
please input item name like idly vada coffee pizza and its quantities from the user and generate the bill
Okay, I can help you with that. Here's a Python code snippet that prompts the user to enter item names and quantities
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more
Variables Terminal
5:38 PM Python 3


AI(3).ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands + Code + Text Run all
total_cost = 0
while True:
item_name = input("Enter item name: ").strip().lower()
if item_name == 'done':
break
if item_name not in item_prices:
print(f"Sorry, '{item_name}' is not on the menu.")
continue
try:
quantity = int(input(f"Enter quantity for {item_name}: ").strip())
if quantity <= 0:
print("Quantity must be a positive number.")
continue
if item_name in items:
items[item_name] += quantity
else:
items[item_name] = quantity
total_cost += item_prices[item_name] * quantity
except ValueError:
print("Invalid quantity. Please enter a number.")
print("\n--- Your Bill ---")
if not items:
print("No items added to the bill.")
else:
for item, quantity in items.items():
price = item_prices[item]
print(f"{item.capitalize()}: {quantity} x ${price:.2f} = ${quantity * price:.2f}")
print("-" * 20)
print(f"Total: ${total_cost:.2f}")
print("-----------------")
# Run the bill generation function
generate_bill()
Gemini
send me a code
Could you please clarify what kind of code you're looking for? For example, are you looking for a specific function, a complete program, or something related to the previous hotel billing system?
please input item name like idly vada coffee pizza and its quantities from the user and generate the bill
Okay, I can help you with that. Here's a Python code snippet that prompts the user to enter item names and quantities and then calculates and displays a simple bill.
I have provided the code to input item names and quantities and generate a basic bill.
Here are some ways you could enhance this billing program:
Add prices for items: Modify the code to include a dictionary or another data structure to store the prices of different items (like idly, vada, coffee, pizza).
Calculate the total cost: Based on the quantities and prices, calculate the total cost of the items and display it on the bill.
Include tax: Add a tax calculation to the total bill.
Implement discounts: Allow for discounts based on certain criteria (e.g., a percentage discount for a minimum total).
Save the bill: Store the generated bill in a file for record-keeping.
Create a simple menu: Display a list of available items and their prices to the user before they start entering their order.
and update the code
Okay, I will update the bill generation code to include item prices and calculate the total cost.
I have updated the code to include item prices and calculate the total bill.
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more

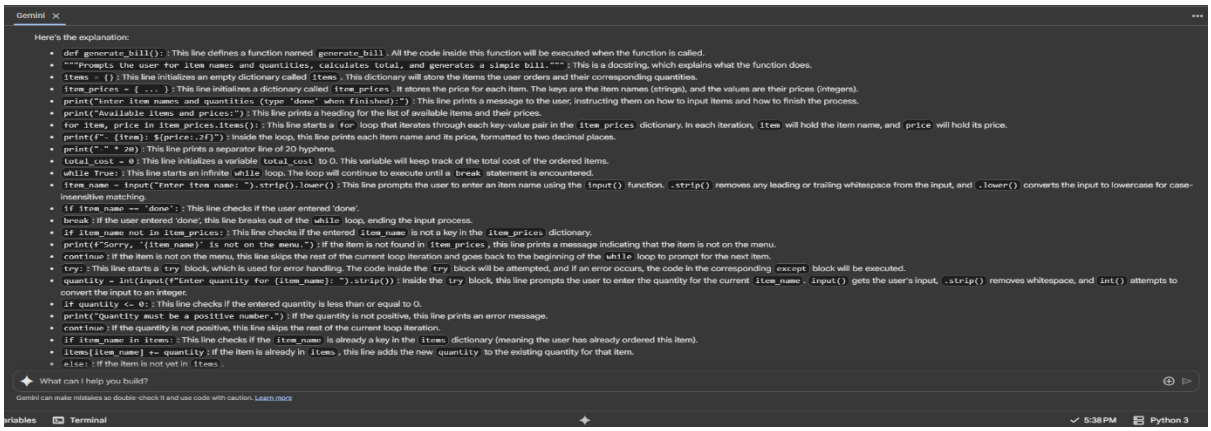

AI(3).ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands + Code + Text Run all
price = item_prices[item]
print(f"{item.capitalize()}: {quantity} x ${price:.2f} = ${quantity * price:.2f}")
print("-" * 20)
print(f"Total: ${total_cost:.2f}")
print("-----------------")
# Run the bill generation function
generate_bill()
Enter item names and quantities (type 'done' when finished):
Available items and prices:
- idly: $10.00
- vada: $12.00
- coffee: $15.00
- pizza: $100.00

Enter item name: idly
Enter quantity for idly: 4
Enter item name: vada
Enter quantity for vada: 2
Enter item name: done
--- Your Bill ---
Idly: 4 x $10.00 = $40.00
Vada: 2 x $12.00 = $24.00

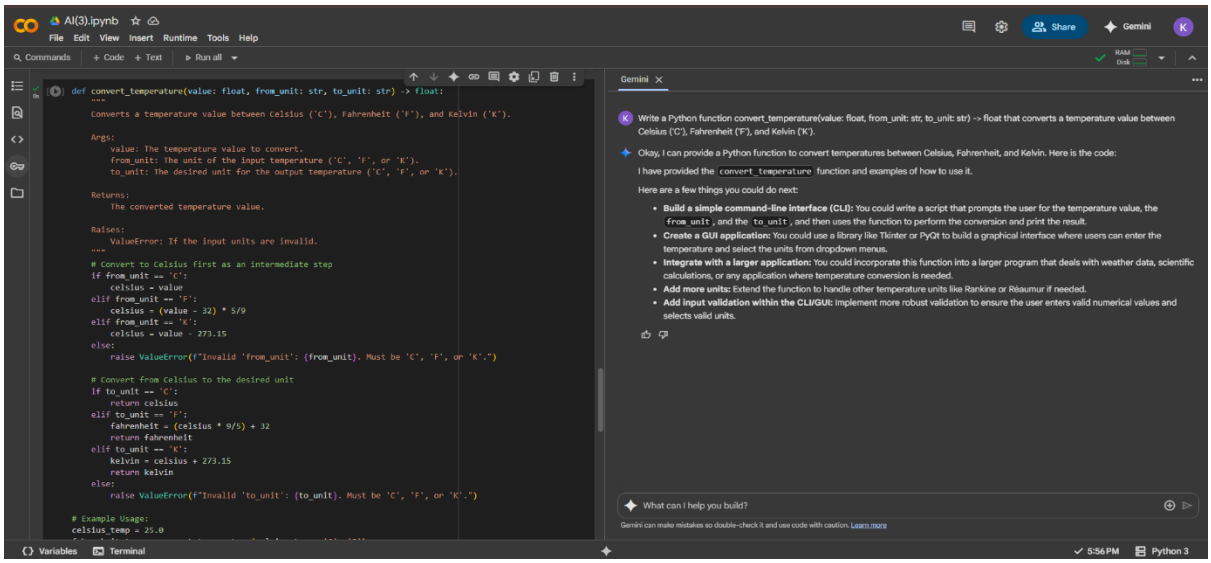
Total: $64.00

Gemini
Implement discounts: Allow for discounts based on certain criteria (e.g., a percentage discount for a minimum total).
Save the bill: Store the generated bill in a file for record-keeping.
Create a simple menu: Display a list of available items and their prices to the user before they start entering their order.
and update the code
Okay, I will update the bill generation code to include item prices and calculate the total cost.
I have updated the code to include item prices and calculate the total bill.
Here are some further improvements or related tasks you could consider:
Add error handling for invalid input: The current code has basic error handling for non-numeric quantity input, but you could add more robust handling for other potential issues.
Implement discounts or taxes: Extend the calculation to include discounts or add tax to the total bill.
Save the bill to a file: Write the generated bill details to a text file or CSV file for record-keeping.
Create a more interactive menu: Instead of just printing the menu, you could present it in a more user-friendly way, perhaps with numbers corresponding to items for easier selection.
Add more items and prices: Expand the item_prices dictionary to include a wider variety of items.
Develop a simple GUI: If you want a visual interface, you could use a library like Tkinter (as in the previous example) to create a graphical billing application.
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more

Explanation:



Task 5:




```
# Example Usage:
celsius_temp = 25.0
fahrenheit_temp = convert_temperature(celsius_temp, 'C', 'F')
print(f"{celsius_temp}°C is {fahrenheit_temp:.2f}°F")

fahrenheit_temp = 77.0
celsius_temp_from_f = convert_temperature(fahrenheit_temp, 'F', 'C')
print(f"{fahrenheit_temp}°F is {celsius_temp_from_f:.2f}°C")

celsius_temp = 0.0
kelvin_temp = convert_temperature(celsius_temp, 'C', 'K')
print(f"{celsius_temp}°C is {kelvin_temp:.2f}K")

kelvin_temp = 300.15
celsius_temp_from_k = convert_temperature(kelvin_temp, 'K', 'C')
print(f"{kelvin_temp}K is {celsius_temp_from_k:.2f}°C")

fahrenheit_temp = 212.0
kelvin_temp_from_f = convert_temperature(fahrenheit_temp, 'F', 'K')
print(f"{fahrenheit_temp}°F is {kelvin_temp_from_f:.2f}K")

kelvin_temp = 373.15
fahrenheit_temp_from_k = convert_temperature(kelvin_temp, 'K', 'F')
print(f"{kelvin_temp}K is {fahrenheit_temp_from_k:.2f}°F")

# Example of invalid unit
try:
    convert_temperature(100, 'X', 'C')
except ValueError as e:
    print(f"Error: {e}")

try:
    convert_temperature(100, 'C', 'Y')
except ValueError as e:
    print(f"Error: {e}")
```

```
25.0°C is 77.00°F
77.0°F is 25.00°C
0.0°C is 273.15K
300.15K is 27.00°C
212.0°F is 373.15K
373.15K is 212.00°F
Error: Invalid 'from_unit': X. Must be 'C', 'F', or 'K'.
Error: Invalid 'to_unit': Y. Must be 'C', 'F', or 'K'.
```

Explanation: