Week - 1 Deliverable Submission:-

Write a simple Python program that:

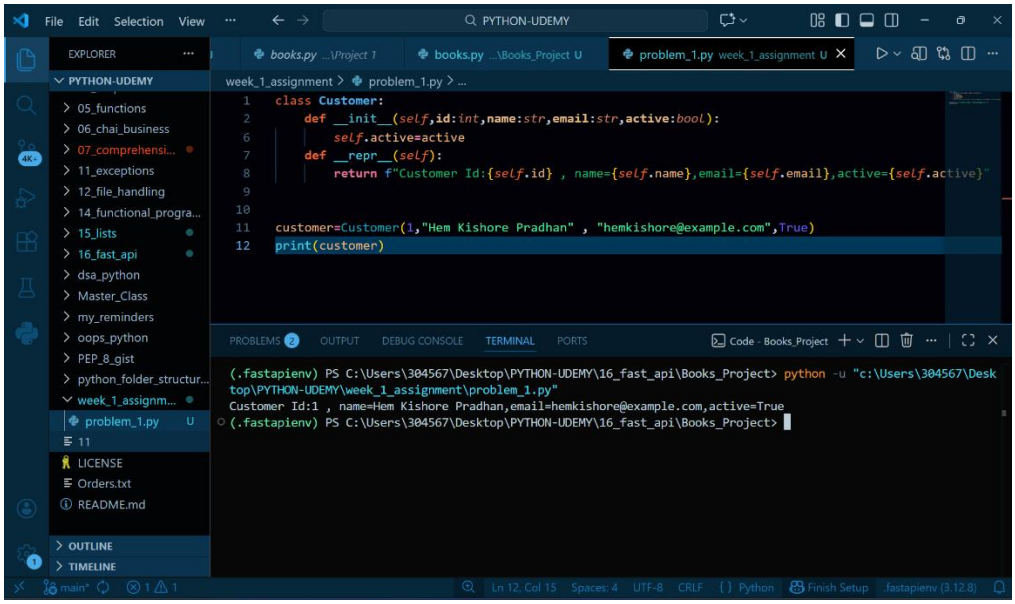
1. Defines a Customer class
   Fields:
   id: int
   name: str
   email: str
   active: bool
   Includes constructor, getters (or direct attributes), and __repr__().

Solution:

```
class Customer:

  def __init__(self,id:int,name:str,email:str,active:bool):

    self.id=id

    self.name=name

    self.email=email

    self.active=active

  def __repr__(self):

    return f"Customer Id:{self.id} ,
name={self.name},email={self.email},active={self.active}"



customer=Customer(1,"Hem Kishore Pradhan" , "hemkishore@example.com",True)
```

Output:

print(customer)

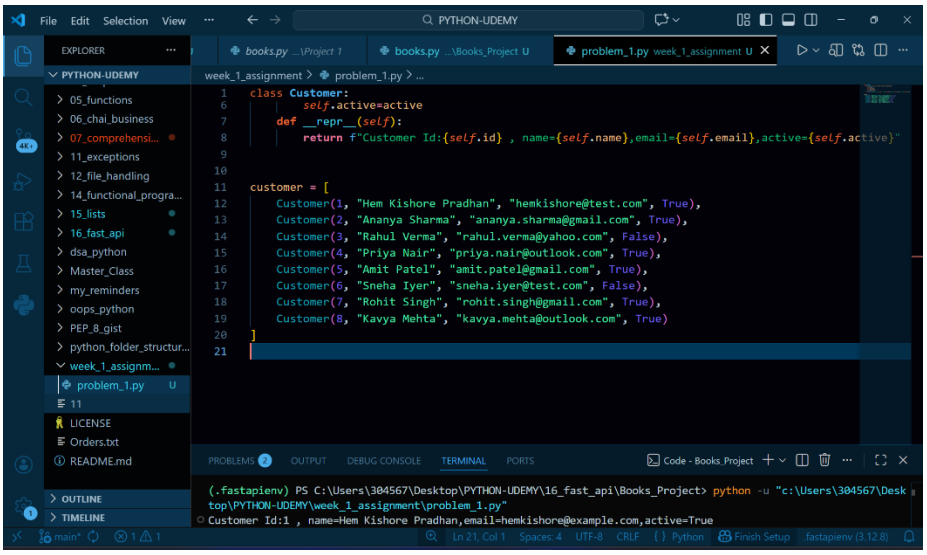


2. Maintains a list of customers

   solution:

```
customer = [
    Customer(1, "Hem Kishore Pradhan", "hemkishore@test.com", True),
    Customer(2, "Ananya Sharma", "ananya.sharma@gmail.com", True),
    Customer(3, "Rahul Verma", "rahul.verma@yahoo.com", False),
    Customer(4, "Priya Nair", "priya.nair@outlook.com", True),
    Customer(5, "Amit Patel", "amit.patel@gmail.com", True),
    Customer(6, "Sneha Iyer", "sneha.iyer@test.com", False),
    Customer(7, "Rohit Singh", "rohit.singh@gmail.com", True),
    Customer(8, "Kavya Mehta", "kavya.mehta@outlook.com", True)
]
```

   Output:



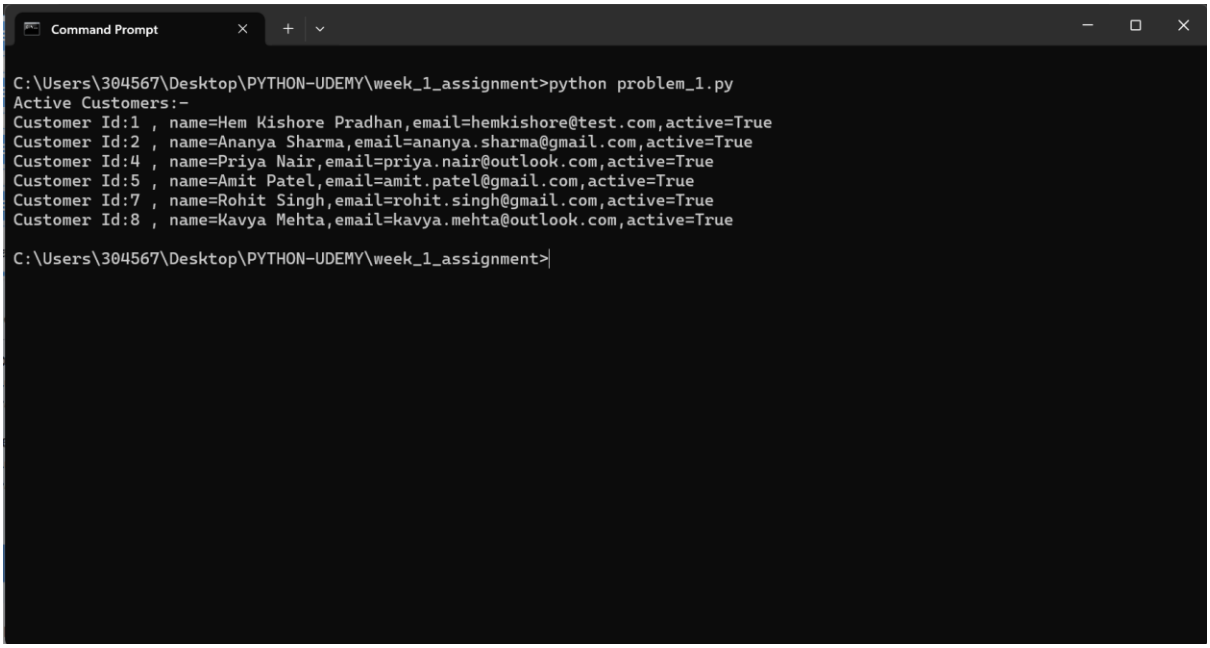
3. Implements data operations using Pythonic patterns

   a) Filter active customers using list comprehension

   Solution:

   active_customers=[c **for** c **in** customer **if** c.active]

   output:

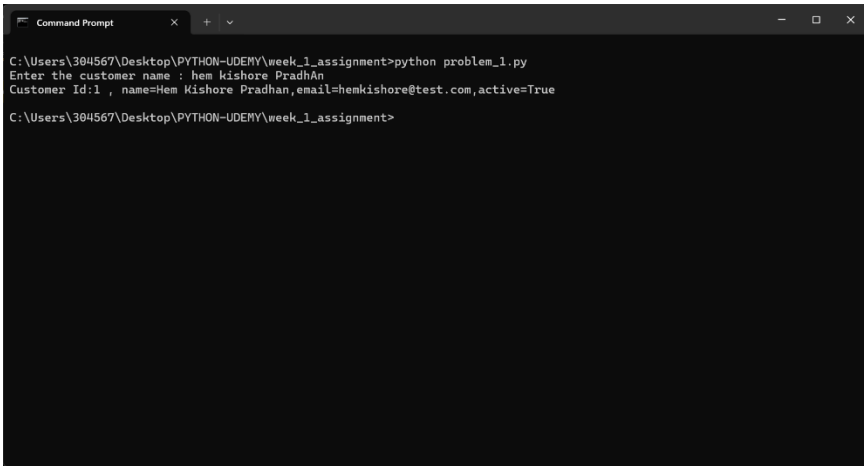


   b) Search customers by name (case-insensitive)

   solution:

```
search_customer_input=str(input("Enter the customer name : "))
for search_customer in customer:
    if search_customer.name.casefold() == search_customer_input.casefold():
        print(search_customer)
```

   output:

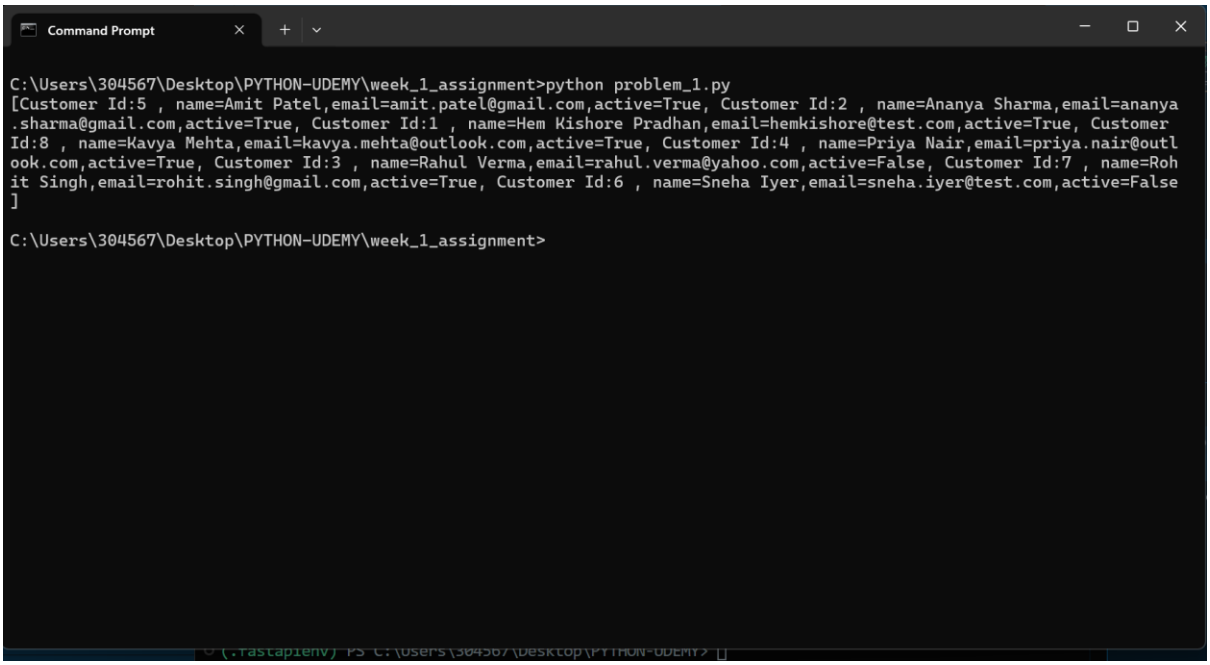


c) Sort customers by name using sorted(key=...)

solution:

sorted_customer=sorted(customer,**key=** lambda **c:**c**.**name)

output:



d) Print results in a clean format

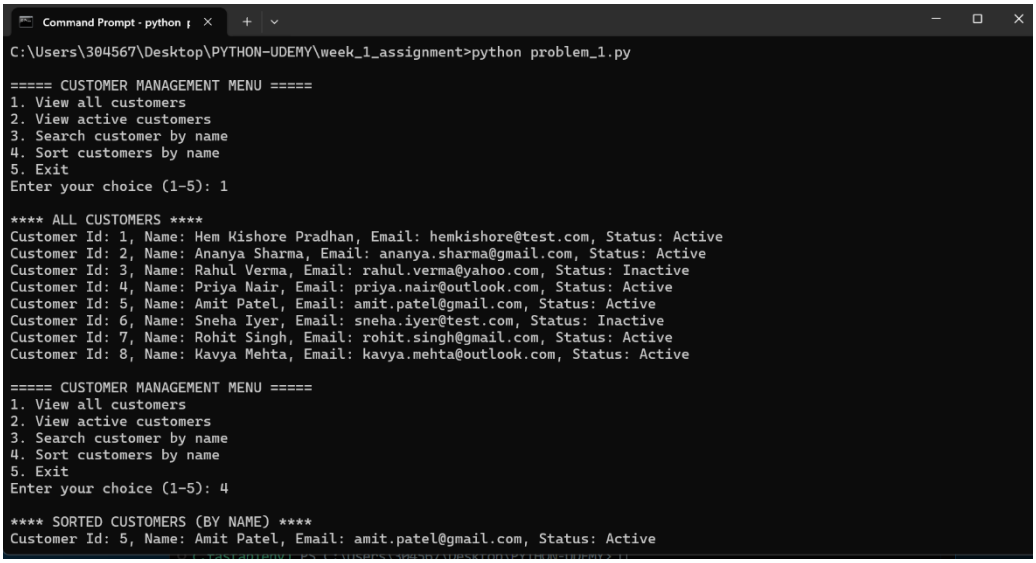
solution:

```
class Customer:
  def __init__(self, id: int, name: str, email: str, active: bool):
    self.id = id
    self.name = name
```

```
        self.email = email
        self.active = active

    def __repr__(self):
        status = "Active" if self.active else "Inactive"
        return f"Customer Id: {self.id}, Name: {self.name}, Email: {self.email},
Status: {status}"

def filter_active_customers(customers):
    return [c for c in customers if c.active]

def search_customer_by_name(name, customers):
    for customer in customers:
        if customer.name.casefold() == name.casefold():
            return customer
    return None

def sort_customers_by_name(customers):
    return sorted(customers, key=lambda c: c.name)

def print_customers(customers):
    if not customers:
        print("No customers found.")
    for c in customers:
        print(c)

# Customer data
customers = [
    Customer(1, "Hem Kishore Pradhan", "hemkishore@test.com", True),
    Customer(2, "Ananya Sharma", "ananya.sharma@gmail.com", True),
    Customer(3, "Rahul Verma", "rahul.verma@yahoo.com", False),
    Customer(4, "Priya Nair", "priya.nair@outlook.com", True),
    Customer(5, "Amit Patel", "amit.patel@gmail.com", True),
    Customer(6, "Sneha Iyer", "sneha.iyer@test.com", False),
    Customer(7, "Rohit Singh", "rohit.singh@gmail.com", True),
    Customer(8, "Kavya Mehta", "kavya.mehta@outlook.com", True)
]

# Menu-driven program
while True:
    print("\n===== CUSTOMER MANAGEMENT MENU =====")
```

```
print("1. View all customers")
print("2. View active customers")
print("3. Search customer by name")
print("4. Sort customers by name")
print("5. Exit")

choice = input("Enter your choice (1-5): ")

if choice == "1":
    print("\n**** ALL CUSTOMERS ****")
    print_customers(customers)

elif choice == "2":
    print("\n**** ACTIVE CUSTOMERS ****")
    active_customers = filter_active_customers(customers)
    print_customers(active_customers)

elif choice == "3":
    name = input("Enter customer name to search: ")
    result = search_customer_by_name(name, customers)
    if result:
        print("\nCustomer Found:")
        print(result)
    else:
        print("\nCustomer not found.")

elif choice == "4":
    print("\n**** SORTED CUSTOMERS (BY NAME) ****")
    sorted_customers = sort_customers_by_name(customers)
    print_customers(sorted_customers)

elif choice == "5":
    print("Exiting program. Goodbye!")
    break

else:
    print("Invalid choice. Please select between 1 and 5.")
```

Output:



4. Write a pseudo-REST contract in comments
   Solution:
   # Pseudo REST API Contract

   # GET /customers
   # Description: Fetch all customers

   # GET /customers?active=True
   # Description: Fetch only active customers

   # GET /customers/search?name=abc
   # Description: Search a customer by name.