MURDOCH
UNIVERSITY
PERTH, WESTERN AUSTRALIA

School of Information Technology

ICT582 ASSIGNMENT
CHECKLIST

Surname (Family Name): ______Subba____

Given Names: ________________Rehar____

Student Number: _____________35126921____

Tutor's Name: ________________Nur Haldar____

Tutorial group: Day _Tuesday_Start time__5:30 PM_Venue__MS Teams_Group ID_ICT582s1_24_____

Assignment Due Date: ___01/06/2024________________**Date Submitted:**___01/06/2024___________

Your assignment should meet the following requirements. Please confirm this (by ticking the boxes) before submitting your assignment.

- ☑ All the details above are complete.
- ☑ The zip file contains the file Assignment.pdf. The documentation was prepared according to the Documentation Requirements specified in the Assignment's question sheet.
- ☑ The zip file includes the code for the three questions. The code for each question is stored under a separate directory named q1, q2, and q3, respectively.
- ☑ You understand that the zip file must be submitted to ICT582 Unit LMS.
- ☑ You have kept a copy of this assignment, including the zip file.

I declare that the work included in this submission is completed independently.

Rehar Subba

Signature ________ ________________________Date ____01/06/2024_____

Table of Contents

I. Introduction

This report delves into a comprehensive discussion of the features developed as part of the ICT582 Major Assignment. It meticulously examines the design and implementation of a simple sales records management system that loads customer and sales data from CSV files into an in-memory structure. Originally, the project set out to incorporate a Graphical User Interface (GUI) using Tkinter for an enriched user experience. However, owing to time limitations, the decision was made to pivot towards a command-line interface (CLI) approach, ensuring adherence to project deadlines while fulfilling the prescribed objectives.

II. Question One: A simple sales records management system

A. Discussion

Initially, I developed a GUI with Tkinter but switched to a CLI due to extensive functionalities and time constraints. For robust data management, I used the os and csv modules. A nested dictionary structure efficiently stores data, enabling easy manipulation. Proper input handling was ensured through while loops and if-else statements. The system accommodates users on different systems for file path handling.

A key feature is saving records as CSV files and maintaining data integrity. Improvements include better error handling, user feedback enhancement, scalability, and transitioning to a GUI for improved usability and experience.

The system's modular design and efficient data storage facilitated smooth operation. While some improvements are needed, the system effectively meets the project's objectives overall.

Strengths and Weaknesses

The function is highly modular, with each task handled by a specific function, making the code easier to debug and extend. Clear prompts and input validations effectively guide the user through the process. The program exhibits flexibility by handling filenames and file paths, ensuring file existence before performing operations. However, the overwrite prompt could be more intuitive by providing additional information about the existing file. The program's error handling could be more robust, especially for file operations. Additionally, the current design is not optimized for large datasets, as all data is stored in memory, which limits scalability.

Improvements

Several improvements can be made to address the program's weaknesses: Implement detailed error messages and exception handling for unexpected situations, such as file access permissions. Enhance the overwrite prompt to show details of the file to be overwritten. Introduce mechanisms to handle large datasets more efficiently, possibly using a database instead of in-memory dictionaries. Transition to a GUI to improve usability and user experience.

B. Self-Diagnosis and Evaluation

Fully Completed and Working Features

- Menu Display and User Input Handling: The menu function displays options and validates user input effectively.
- Loading Records: Records are correctly loaded from CSV files into a nested dictionary in memory.
- Saving Customer Records: Customer records can be saved to a specified file with the correct handling of overwriting.
- Saving Sales Records: Sales records can be saved similarly, ensuring data integrity.
- Printing Records (Head-like function): The print_dict function successfully prints a specified number of records.

Completed but Not Fully Working Features

- File Path Handling: While functional, the file path handling could be more robust and informative, particularly for incorrect paths. We could go through more test cases to improve this.

Not Fully Completed or Not Attempted Features

- Advanced Error Handling: More comprehensive exception handling is needed for unexpected errors during file operations and data processing.
- GUI: A GUI is needed to make the system more intuitive. A command line interface may not be very user-friendly and may be error-prone.

C. Test Evidence

- Feature: Load records to memory.

  Test Case: The Data structure is empty when selecting option 1 (Loading records to memory).

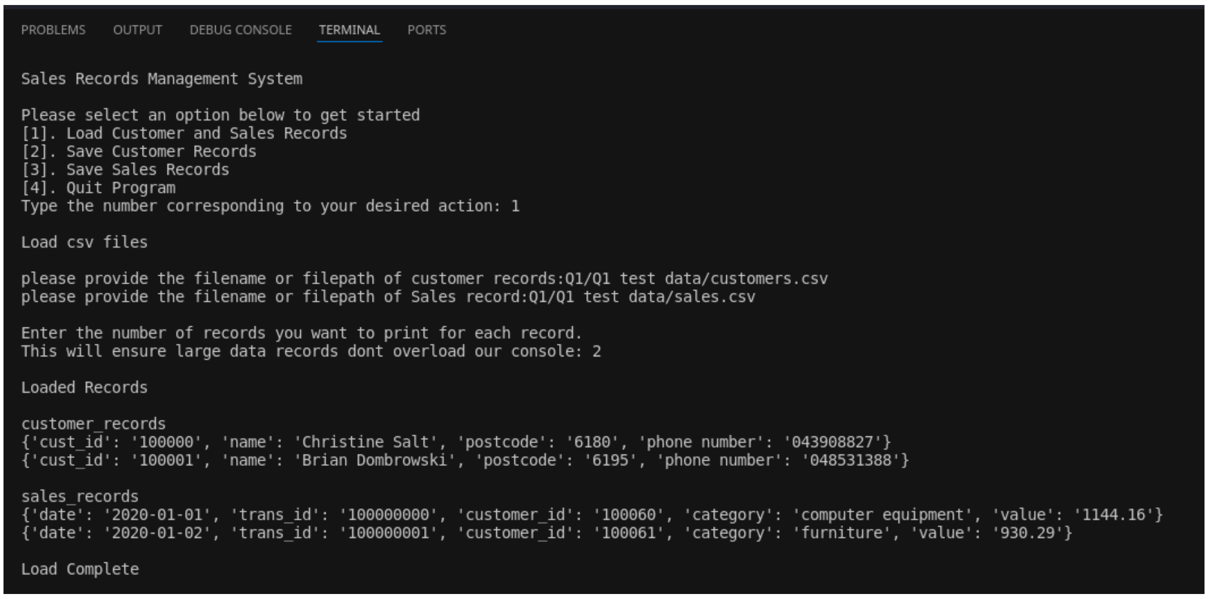
  Functions utilized: Option1 and load_records

  Arguments passed:

  Q1/Q1 test data/customers.csv as customer record

  Q1/Q1 test data/sales.csv as a sales record.

  2 as the number of lines passed to the print_dict function.



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 1

Load csv files

please provide the filename or filepath of customer records:Q1/Q1 test data/customers.csv
please provide the filename or filepath of Sales record:Q1/Q1 test data/sales.csv

Enter the number of records you want to print for each record.
This will ensure large data records dont overload our console: 2

Loaded Records

customer_records
{'cust_id': '100000', 'name': 'Christine Salt', 'postcode': '6180', 'phone number': '043908827'}
{'cust_id': '100001', 'name': 'Brian Dombrowski', 'postcode': '6195', 'phone number': '048531388'}

sales_records
{'date': '2020-01-01', 'trans_id': '100000000', 'customer_id': '100060', 'category': 'computer equipment', 'value': '1144.16'}
{'date': '2020-01-02', 'trans_id': '100000001', 'customer_id': '100061', 'category': 'furniture', 'value': '930.29'}

Load Complete
```

Conclusion: Feature meets requirement.

- Feature: Load records to memory.

  Test Case: The Data structure is not empty when selecting option 1 (Loading records to memory).

  Functions utilized: Option1 and load_records

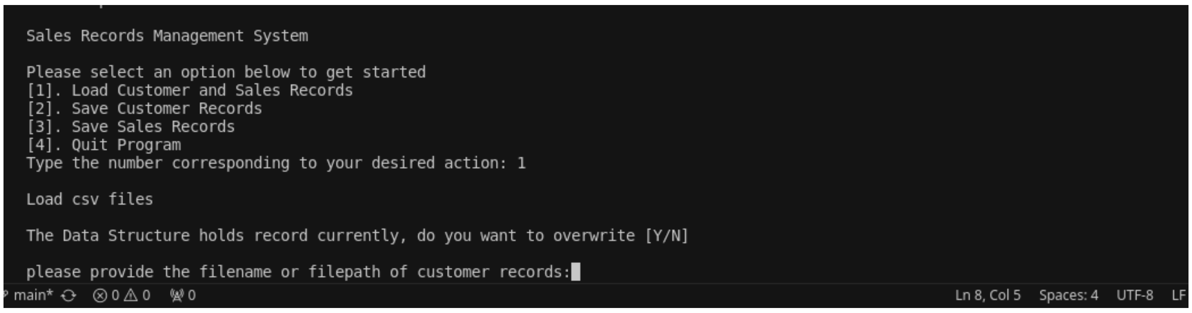
  Arguments passed:

  Q1/Q1 test data/customers.csv as customer record

  Q1/Q1 test data/sales.csv as a sales record.

  Overwrite: Y/y or just enter

  2 as the number of lines passed to the print_dict function.



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 1

Load csv files

The Data Structure holds record currently, do you want to overwrite [Y/N]

please provide the filename or filepath of customer records:
main*   ⊗0 ⚠0   ⑆0                                  Ln 8, Col 5   Spaces: 4   UTF-8   LF
```

Conclusion: Feature meets requirement.

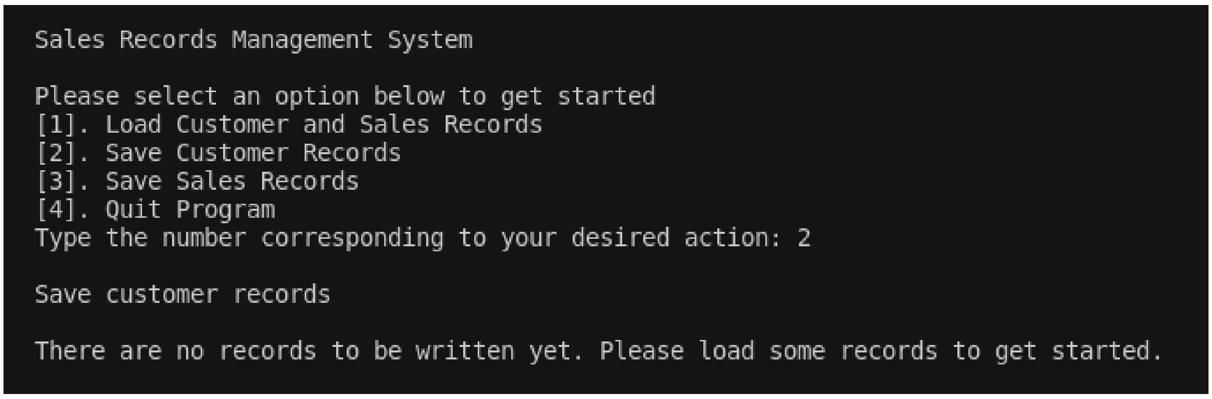
- Feature: save customer records

  Test Case: The Data structure is empty when selecting option 2 (saving customer records from memory to file)

  Functions utilized: Option2, csv_writer and save_records

  Arguments passed:

  None



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 2

Save customer records

There are no records to be written yet. Please load some records to get started.
```

Conclusion: Feature meets requirement.

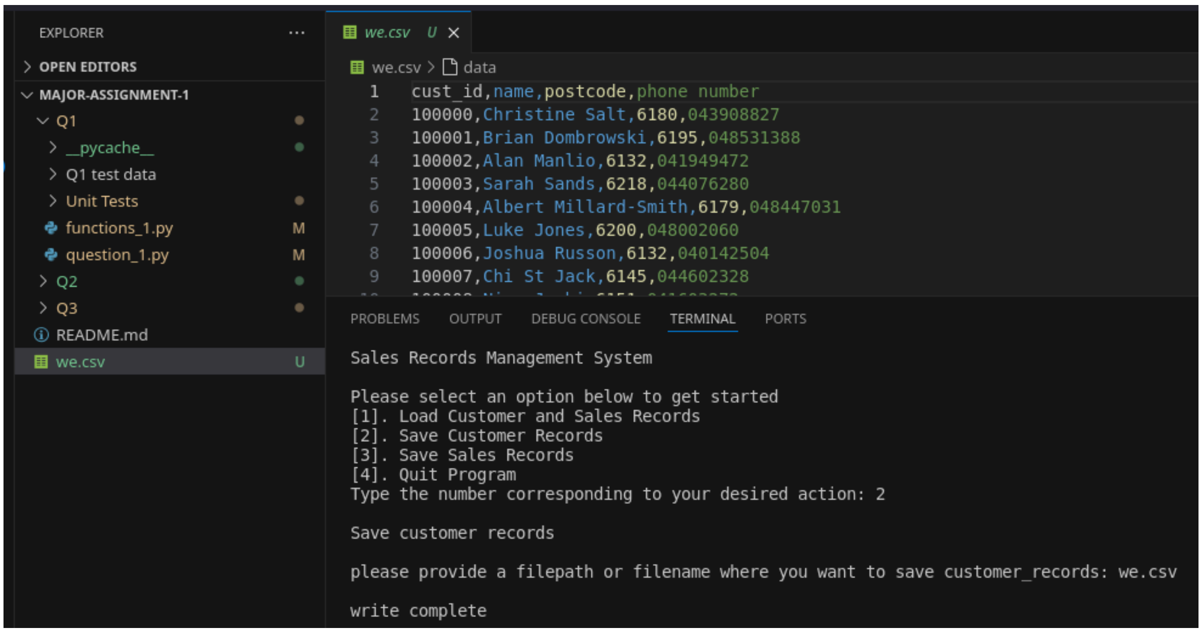
- Feature: save customer records

  Test Case: The Data structure is not empty when selecting option 2 (saving customer records from memory to file).

  Functions utilized: Option2, csv_writer and save_records

  Arguments passed:

  we.csv as filename



EXPLORER

OPEN EDITORS

MAJOR-ASSIGNMENT-1

we.csv > data

```
cust_id,name,postcode,phone number
100000,Christine Salt,6180,043908827
100001,Brian Dombrowski,6195,048531388
100002,Alan Manlio,6132,041949472
100003,Sarah Sands,6218,044076280
100004,Albert Millard-Smith,6179,048447031
100005,Luke Jones,6200,048002060
100006,Joshua Russon,6132,040142504
100007,Chi St Jack,6145,044602328
```

```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 2

Save customer records

please provide a filepath or filename where you want to save customer_records: we.csv

write complete
```

Conclusion: Feature meets requirement.

- Feature: save customer records

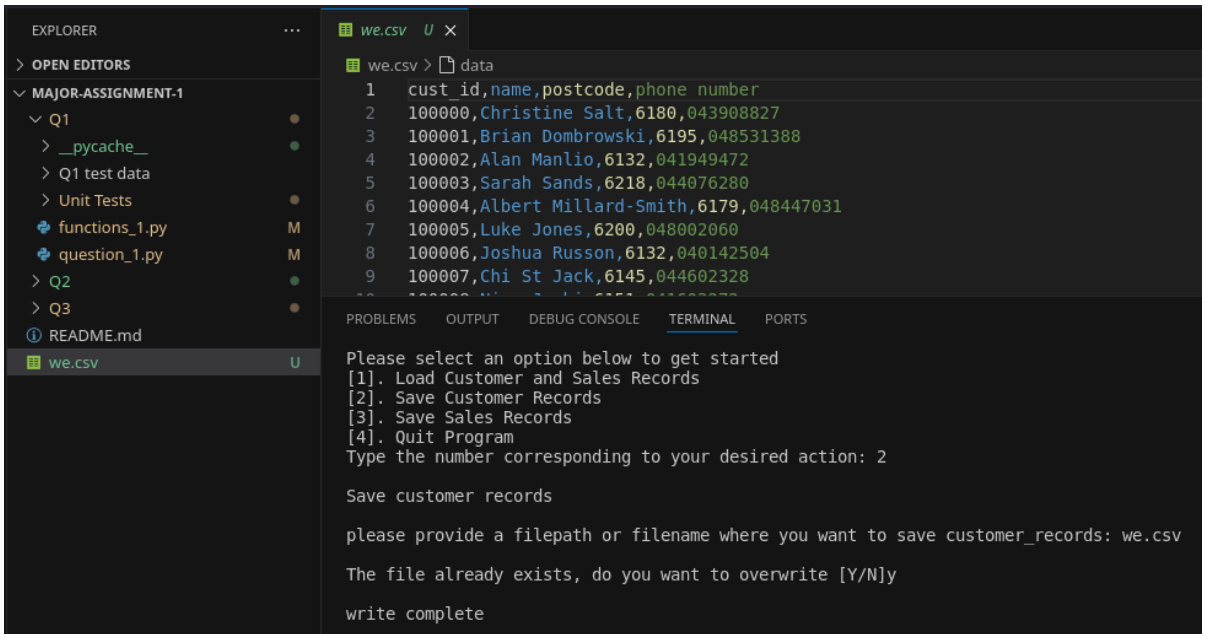
  Test Case: The filename provided exists when selecting option 2 (saving customer records from memory to file).

  Functions utilized: Option2, csv_writer and save_records

  Arguments passed:

  we.csv as filename

  Overwrite: Y, y or enter.



EXPLORER

OPEN EDITORS

MAJOR-ASSIGNMENT-1

we.csv > data

```
cust_id,name,postcode,phone number
100000,Christine Salt,6180,043908827
100001,Brian Dombrowski,6195,048531388
100002,Alan Manlio,6132,041949472
100003,Sarah Sands,6218,044076280
100004,Albert Millard-Smith,6179,048447031
100005,Luke Jones,6200,048002060
100006,Joshua Russon,6132,040142504
100007,Chi St Jack,6145,044602328
```

```
Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 2

Save customer records

please provide a filepath or filename where you want to save customer_records: we.csv

The file already exists, do you want to overwrite [Y/N]y

write complete
```

Conclusion: Feature meets requirement.

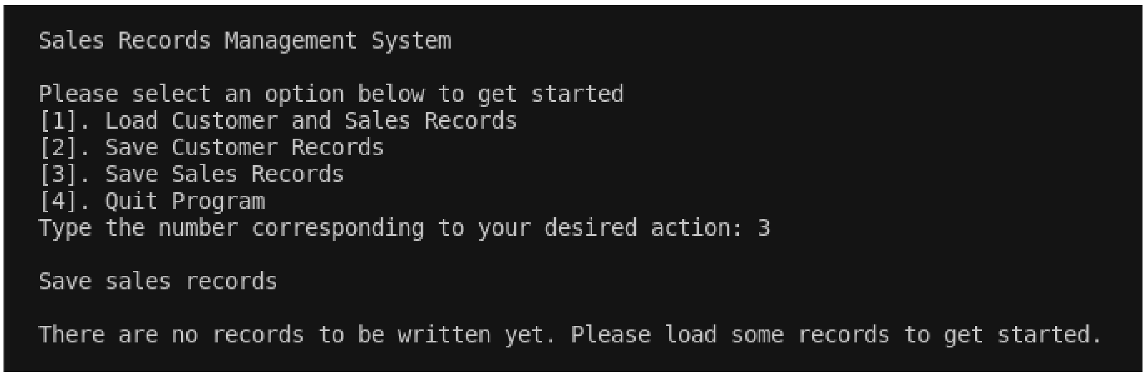
- Feature: save sales records

  Test Case: The Data structure is empty when selecting option 3 (saving sales records from memory to file)

  Functions utilized: Option3, csv_writer and save_records

  Arguments passed:

  None



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 3

Save sales records

There are no records to be written yet. Please load some records to get started.
```

Conclusion: Feature meets requirement.

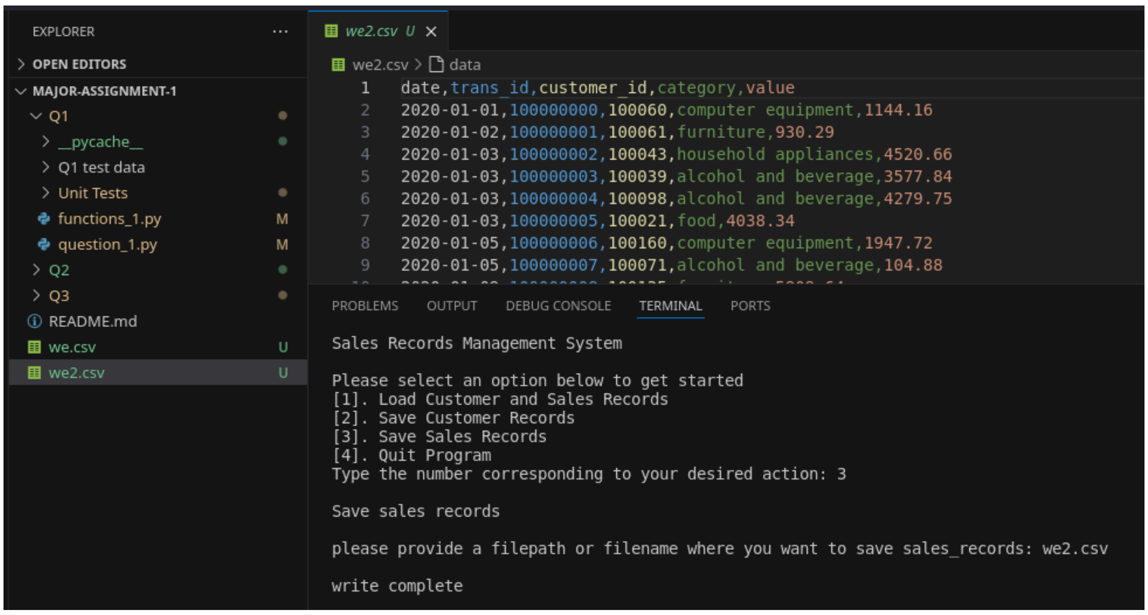
- Feature: save sales records

  Test Case: The Data structure is not empty when selecting option 3 (saving sales records from memory to file)

  Functions utilized: Option3, csv_writer and save_records

  Arguments passed:

  we2.csv as filename



```
date,trans_id,customer_id,category,value
2020-01-01,100000000,100060,computer equipment,1144.16
2020-01-02,100000001,100061,furniture,930.29
2020-01-03,100000002,100043,household appliances,4520.66
2020-01-03,100000003,100039,alcohol and beverage,3577.84
2020-01-03,100000004,100098,alcohol and beverage,4279.75
2020-01-03,100000005,100021,food,4038.34
2020-01-05,100000006,100160,computer equipment,1947.72
2020-01-05,100000007,100071,alcohol and beverage,104.88
```

```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 3

Save sales records

please provide a filepath or filename where you want to save sales_records: we2.csv

write complete
```

Conclusion: Feature meets requirement.

- Feature: save sales records

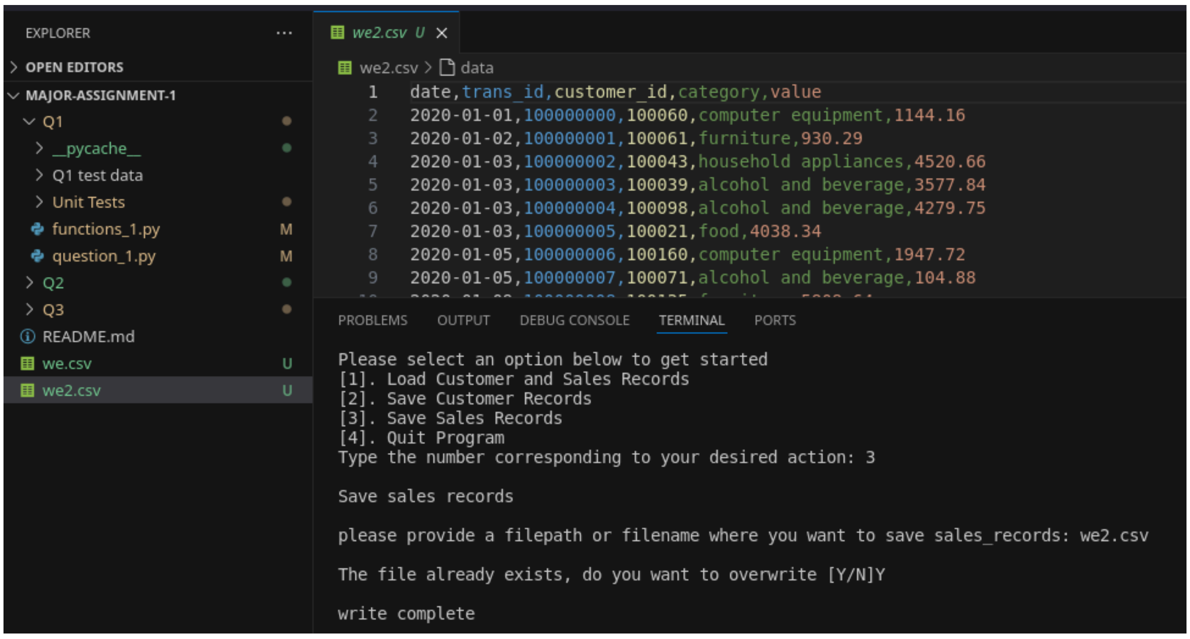
  Test Case: The filename provided exists when selecting option 3 (saving customer records from memory to file).

  Functions utilized: Option3, csv_writer and save_records

  Arguments passed:

    we2.csv as filename

    Overwrite: Y, y or enter.



```
date,trans_id,customer_id,category,value
2020-01-01,100000000,100060,computer equipment,1144.16
2020-01-02,100000001,100061,furniture,930.29
2020-01-03,100000002,100043,household appliances,4520.66
2020-01-03,100000003,100039,alcohol and beverage,3577.84
2020-01-03,100000004,100098,alcohol and beverage,4279.75
2020-01-03,100000005,100021,food,4038.34
2020-01-05,100000006,100160,computer equipment,1947.72
2020-01-05,100000007,100071,alcohol and beverage,104.88
```

```
Please select an option below to get started
[1]. Load Customer and Sales Records
[2]. Save Customer Records
[3]. Save Sales Records
[4]. Quit Program
Type the number corresponding to your desired action: 3

Save sales records

please provide a filepath or filename where you want to save sales_records: we2.csv

The file already exists, do you want to overwrite [Y/N]Y

write complete
```

Conclusion: Feature meets requirement.

D. Filenames:

- question_1.py: Contains the main program logic and menu handling.
- functions_1.py: Contains the functions for loading, saving, and printing records.
- Q1 test data (Dir):
  - customers.csv: Example CSV file for customer records.
  - sales.csv: Example CSV file for sales records.
- Unit tests (Dir): Contains unit tests with sample data

III. Question 2: Search and manipulation of customer and sales records

A. Discussion of the Solution

The Sales Records Management System employs an iterative ID assignment strategy, ensuring sequential alignment and uniqueness within the dataset. This method prioritizes simplicity and consistency, facilitating easy management and maintaining data integrity. The system's design includes a large ID generation range to anticipate future scalability, although adjustments may become necessary as the dataset expands beyond the given range. Strategies for optimizing ID generation to accommodate larger datasets while preserving uniqueness and sequential alignment are vital for long-term scalability.

Strength and Weaknesses

In terms of strengths, the system features a modular structure that enhances readability and maintainability by organizing functionality into distinct, manageable components. Furthermore, its sequential and unique ID generation mechanism bolsters data integrity by assigning each record a distinct identifier. However, the system's error handling could be more robust, particularly for edge cases that may arise during operation. Additionally, redundancy in certain parts of the code, especially in handling file operations and user input, presents an opportunity for improvement. While basic input validation is implemented, employing more sophisticated techniques for validating phone numbers and dates could enhance data integrity. Moreover, hardcoded elements such as headers and file paths limit the program's flexibility, and making them dynamic or configurable would increase its adaptability to different environments.

Improvements

Several improvements are recommended to enhance the sales records management system. Implementing robust error handling and providing clear exception messages would enhance system reliability and user-friendliness. Secondly, transitioning to a database system for data storage could efficiently handle larger datasets and improve scalability. Additionally, using more sophisticated algorithms for ID generation could enhance scalability and performance. Refactoring the code to reduce redundancy and improve maintainability, implementing enhanced validation for inputs, and introducing a configuration file to manage constants would further enhance the system's flexibility and usability. Addressing these areas would make the system more efficient, reliable, and user-friendly.

B. Self-Diagnosis and Evaluation

Fully Completed and Working Features:

- Adding new customer details.
- Adding new sales records for existing customers.
- Searching customers and sales records using a single search string.
- Displaying all sales records from a customer using their customer ID.
- Deleting sales records with a given transaction ID.
- Deleting a customer with a given customer ID and all associated sales records.

Completed but Not Fully Working Features:

- None.

Not Fully Completed or Not Attempted Features:

- All features have been fully implemented and tested.

C. Test Evidence

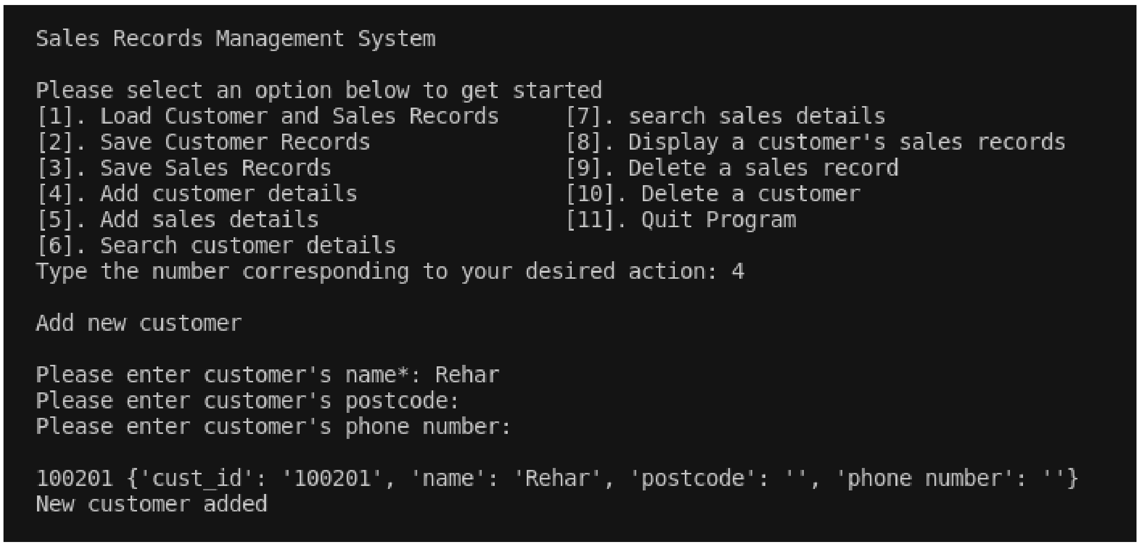
- Feature: Add a new customer

  Test Case: Name is entered for Option 4 (Adding Customer details)

  Functions utilized: Option4

  Arguments passed:

      Rehar as Name
      empty phone number
      empty postcode



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 4

Add new customer

Please enter customer's name*: Rehar
Please enter customer's postcode:
Please enter customer's phone number:

100201 {'cust_id': '100201', 'name': 'Rehar', 'postcode': '', 'phone number': ''}
New customer added
```

Conclusion: Feature meets requirement.

- Feature: Add a new customer

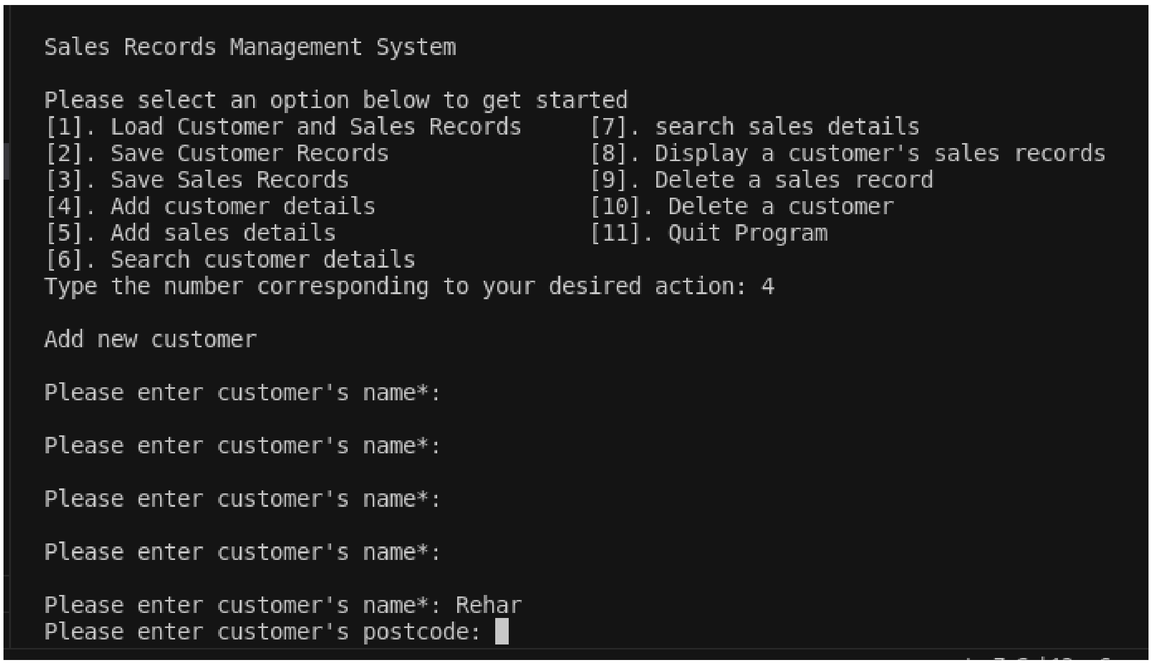
  Test Case: Name is empty for Option 4 (Adding Customer details)

  Functions utilized: Option4

  Arguments passed:

  empty name field

  Expected result: Loop until the name is entered.



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 4

Add new customer

Please enter customer's name*:

Please enter customer's name*:

Please enter customer's name*:

Please enter customer's name*:

Please enter customer's name*: Rehar
Please enter customer's postcode:
```

Conclusion: Feature meets requirement.

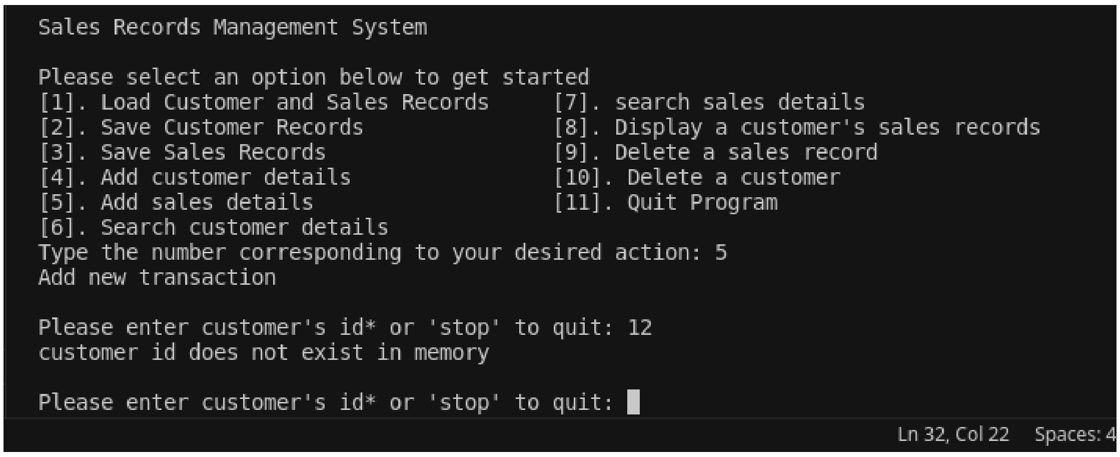
- Feature: Add a new sales record

  Test Case: customer ID does not exist for Option 5 (Adding sales details)

  Functions utilized: Option5

  Arguments passed:

  12 as customer ID



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 5
Add new transaction

Please enter customer's id* or 'stop' to quit: 12
customer id does not exist in memory

Please enter customer's id* or 'stop' to quit: 
```

Conclusion: Feature meets requirement.

- Feature: Add a new sales record

  Test Case: Customer ID does exist for Option 5 (Adding sales details)

  Functions utilized: Option5

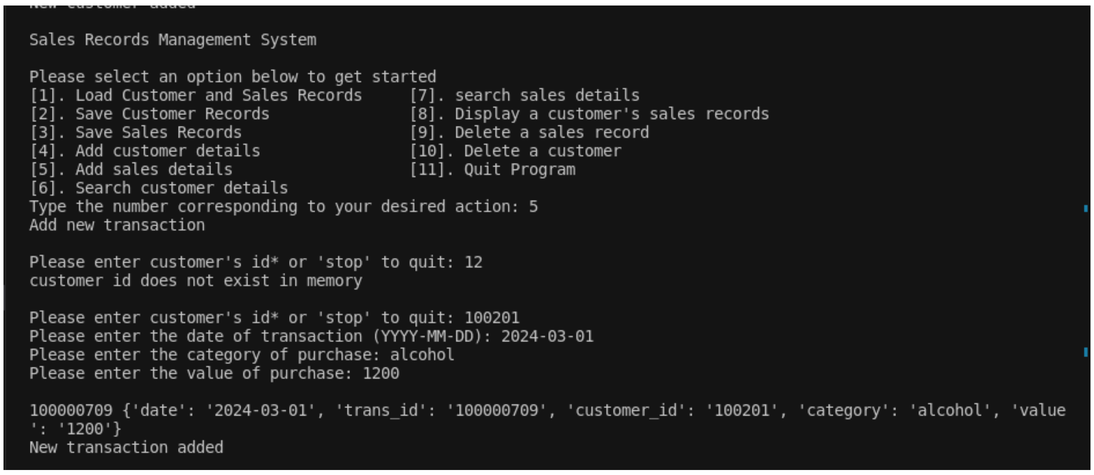
  Arguments passed:

  100201 as customer ID

  2024-03-01 as date

  alcohol as category

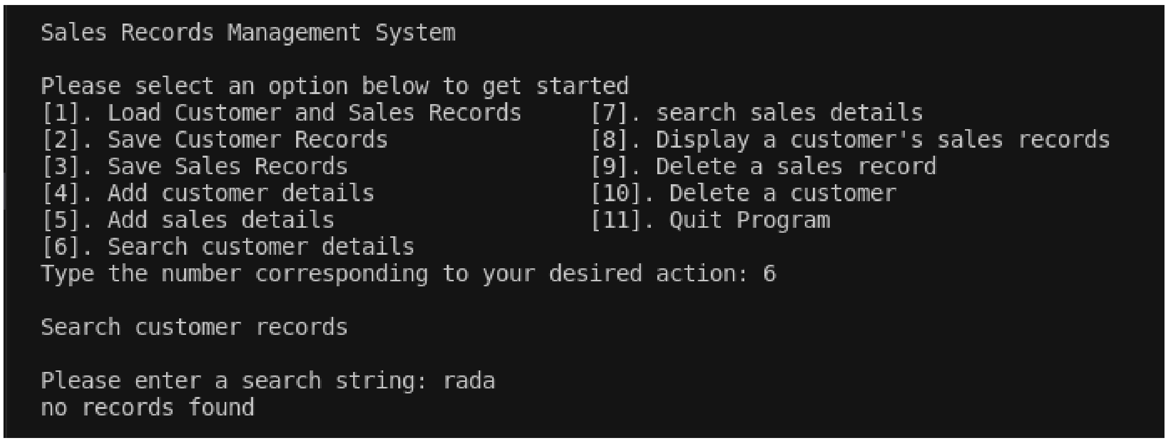
  1200 as value



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 5
Add new transaction

Please enter customer's id* or 'stop' to quit: 12
customer id does not exist in memory

Please enter customer's id* or 'stop' to quit: 100201
Please enter the date of transaction (YYYY-MM-DD): 2024-03-01
Please enter the category of purchase: alcohol
Please enter the value of purchase: 1200

100000709 {'date': '2024-03-01', 'trans_id': '100000709', 'customer_id': '100201', 'category': 'alcohol', 'value
': '1200'}
New transaction added
```

Conclusion: Feature meets requirement.

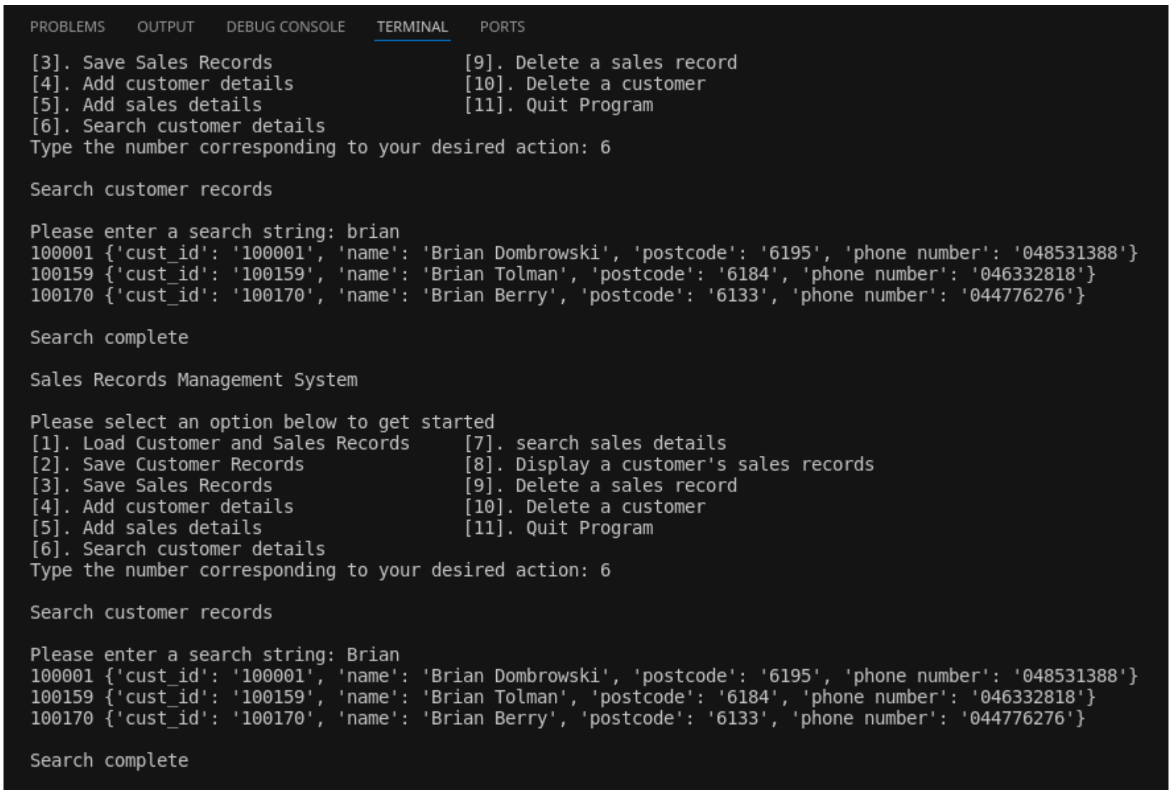
- Feature: Search customers using a single case-insensitive string, which allows partial matches.
  Test Case: The search string does not exist in the data for Option 6 (searching customer details)
  Functions utilized: Option6
  Arguments passed:
  Rada as the search string



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 6

Search customer records

Please enter a search string: rada
no records found
```

Conclusion: Feature meets requirement.

- Feature: Search customers using a single case-insensitive string, which allows partial matches.
  Test Case: search string exists in data for Option 6 (searching customer details)
  Functions utilized: Option6
  Arguments passed:
  Brian, as the search string
  brian, as the search string



```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
[3]. Save Sales Records                 [9]. Delete a sales record
[4]. Add customer details               [10]. Delete a customer
[5]. Add sales details                  [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 6

Search customer records

Please enter a search string: brian
100001 {'cust_id': '100001', 'name': 'Brian Dombrowski', 'postcode': '6195', 'phone number': '048531388'}
100159 {'cust_id': '100159', 'name': 'Brian Tolman', 'postcode': '6184', 'phone number': '046332818'}
100170 {'cust_id': '100170', 'name': 'Brian Berry', 'postcode': '6133', 'phone number': '044776276'}

Search complete

Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records    [7]. search sales details
[2]. Save Customer Records              [8]. Display a customer's sales records
[3]. Save Sales Records                 [9]. Delete a sales record
[4]. Add customer details               [10]. Delete a customer
[5]. Add sales details                  [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 6

Search customer records

Please enter a search string: Brian
100001 {'cust_id': '100001', 'name': 'Brian Dombrowski', 'postcode': '6195', 'phone number': '048531388'}
100159 {'cust_id': '100159', 'name': 'Brian Tolman', 'postcode': '6184', 'phone number': '046332818'}
100170 {'cust_id': '100170', 'name': 'Brian Berry', 'postcode': '6133', 'phone number': '044776276'}

Search complete
```

Conclusion: Feature meets requirement.

- Feature: Search sales records using a single case-insensitive string, which allows partial matches.

  Test Case: search string exists in data for Option 7 (searching sales details)

  Functions utilized: Option7

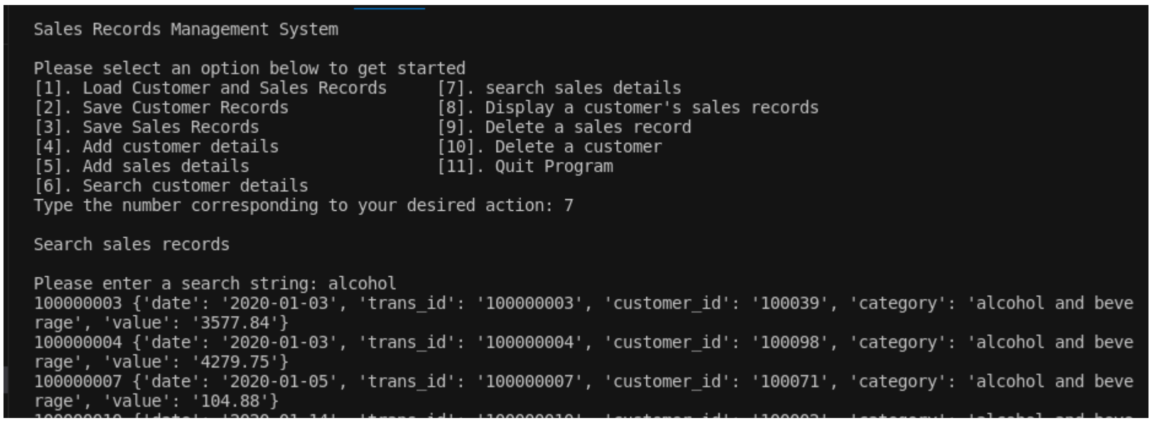
  Arguments passed:

  alcohol as the search string

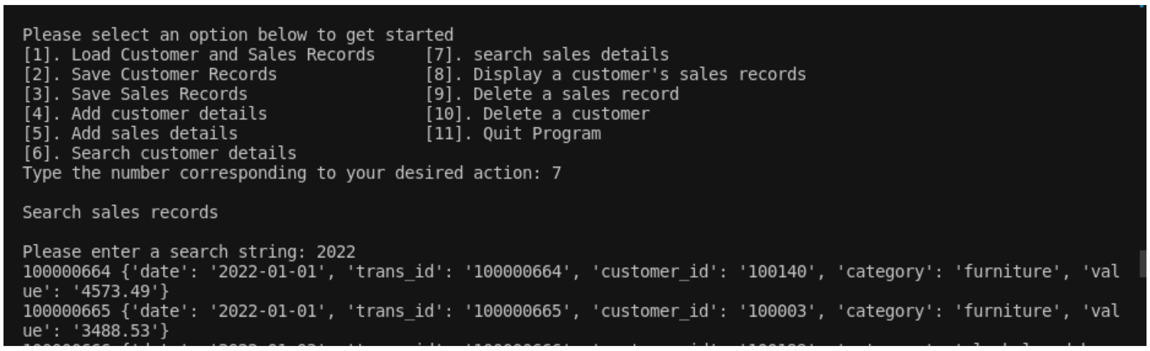
  2022 as the search string

  100003 as the search string

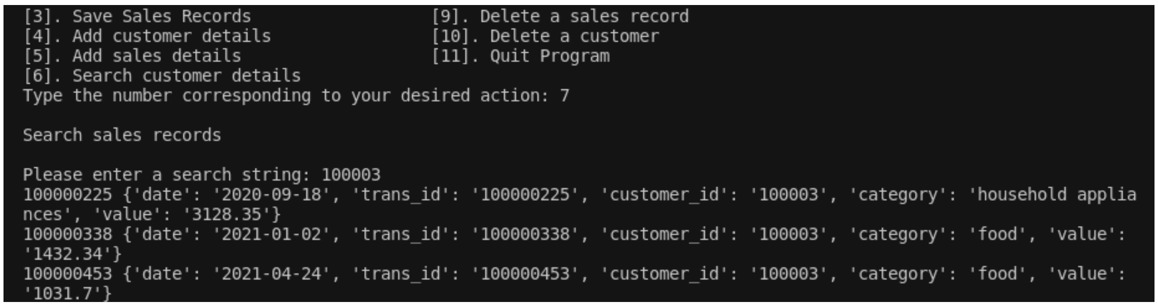
  1200 as the search string



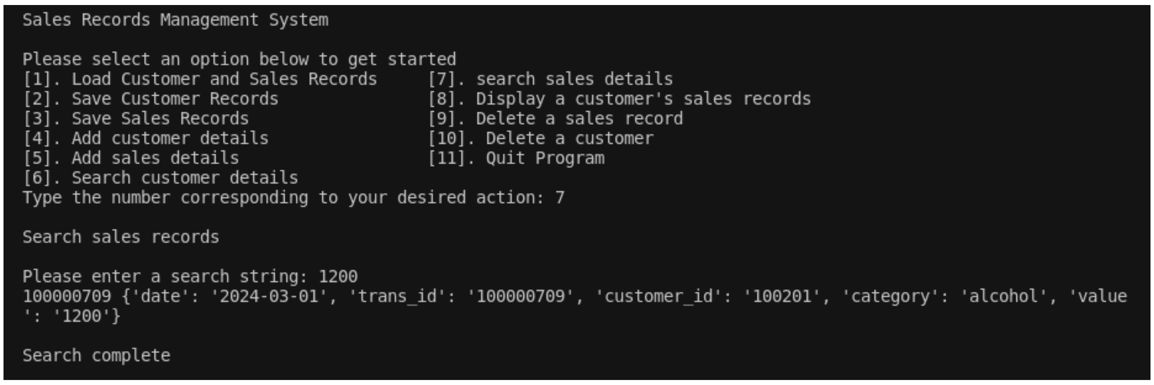
```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records    [7]. search sales details
[2]. Save Customer Records              [8]. Display a customer's sales records
[3]. Save Sales Records                 [9]. Delete a sales record
[4]. Add customer details               [10]. Delete a customer
[5]. Add sales details                  [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 7

Search sales records

Please enter a search string: alcohol
100000003 {'date': '2020-01-03', 'trans_id': '100000003', 'customer_id': '100039', 'category': 'alcohol and beve
rage', 'value': '3577.84'}
100000004 {'date': '2020-01-03', 'trans_id': '100000004', 'customer_id': '100098', 'category': 'alcohol and beve
rage', 'value': '4279.75'}
100000007 {'date': '2020-01-05', 'trans_id': '100000007', 'customer_id': '100071', 'category': 'alcohol and beve
rage', 'value': '104.88'}
```



```
Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 7

Search sales records

Please enter a search string: 2022
100000664 {'date': '2022-01-01', 'trans_id': '100000664', 'customer_id': '100140', 'category': 'furniture', 'val
ue': '4573.49'}
100000665 {'date': '2022-01-01', 'trans_id': '100000665', 'customer_id': '100003', 'category': 'furniture', 'val
ue': '3488.53'}
```



```
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 7

Search sales records

Please enter a search string: 100003
100000225 {'date': '2020-09-18', 'trans_id': '100000225', 'customer_id': '100003', 'category': 'household applia
nces', 'value': '3128.35'}
100000338 {'date': '2021-01-02', 'trans_id': '100000338', 'customer_id': '100003', 'category': 'food', 'value':
'1432.34'}
100000453 {'date': '2021-04-24', 'trans_id': '100000453', 'customer_id': '100003', 'category': 'food', 'value':
'1031.7'}
```



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 7

Search sales records

Please enter a search string: 1200
100000709 {'date': '2024-03-01', 'trans_id': '100000709', 'customer_id': '100201', 'category': 'alcohol', 'value
': '1200'}

Search complete
```

Conclusion: Feature meets requirement.

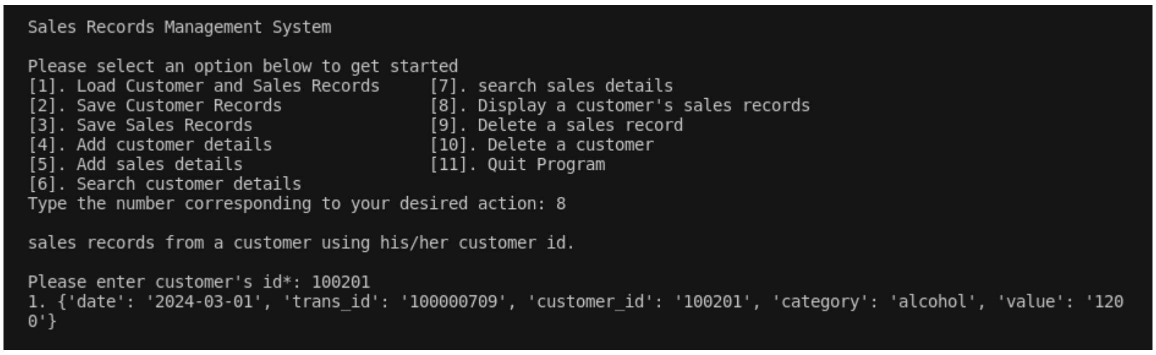
- Feature: Display all sales records from a customer using his/her customer-id

  Test Case: Customer ID exists in data for Option 8, and the customer has transactions (searching sales record based on customer ID)

  Functions utilized: Option8

  Arguments passed:

  100201 as customer ID



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 8

sales records from a customer using his/her customer id.

Please enter customer's id*: 100201
1. {'date': '2024-03-01', 'trans_id': '100000709', 'customer_id': '100201', 'category': 'alcohol', 'value': '120
0'}
```

Conclusion: Feature meets requirement.

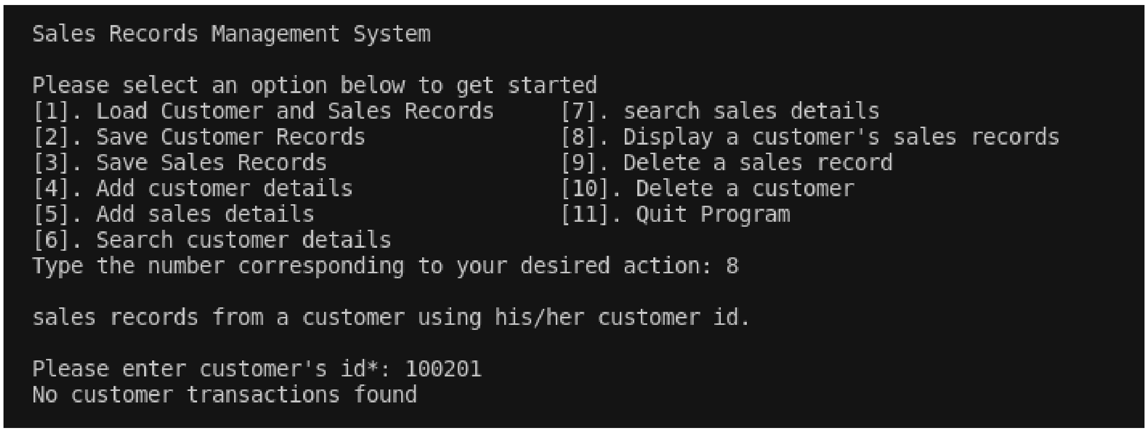
- Feature: Display all sales records from a customer using his/her customer-id

  Test Case: Customer ID exists in data for Option 8, but the customer does not have transactions (searching sales record based on customer ID)

  Functions utilized: Option8

  Arguments passed:

  100201 as customer ID



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 8

sales records from a customer using his/her customer id.

Please enter customer's id*: 100201
No customer transactions found
```

Conclusion: Feature meets requirement.

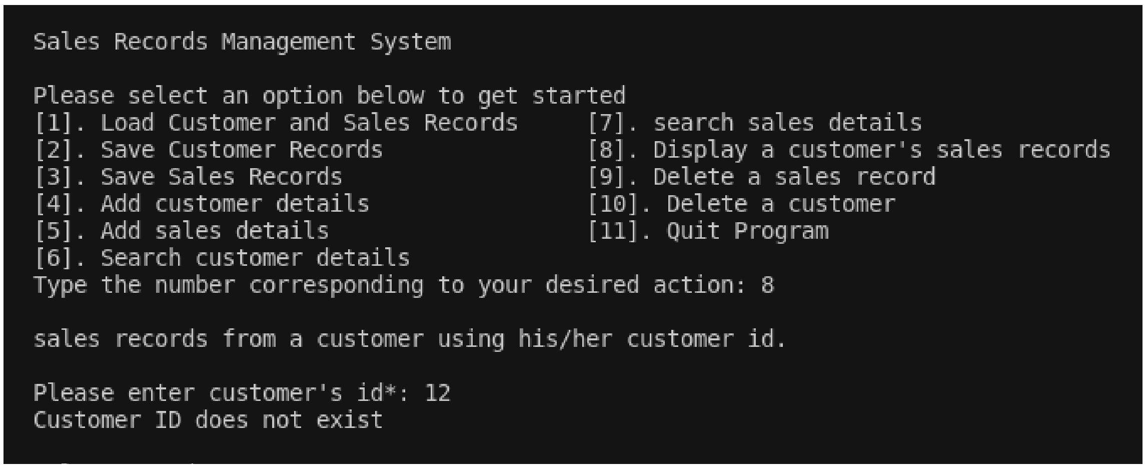
- Feature: Display all sales records from a customer using his/her customer-id

  Test Case: Customer ID does not exist in data for Option 8 (searching sales record based on customer ID)

  Functions utilized: Option8

  Arguments passed:

  12 as customer ID



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 8

sales records from a customer using his/her customer id.

Please enter customer's id*: 12
Customer ID does not exist
```

Conclusion: Feature meets requirement.

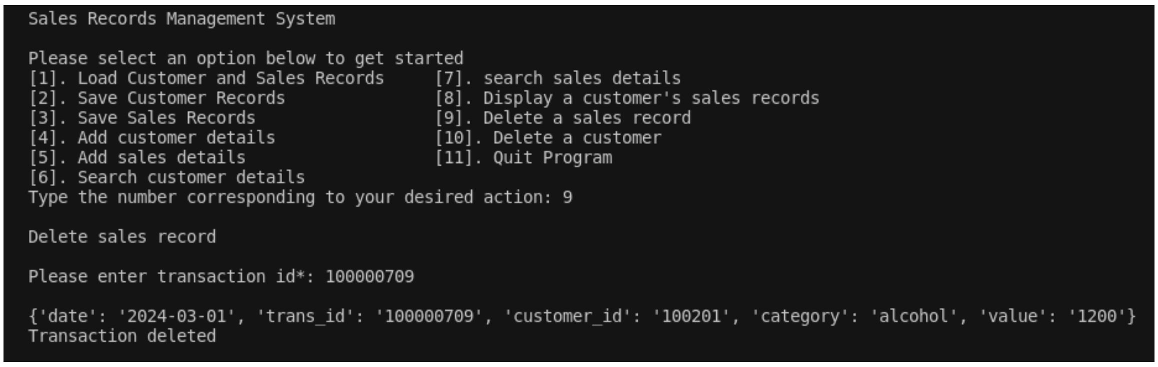
- Feature: Delete a sales record with transaction ID

  Test Case: Transaction exists in data for Option 9 (deleting a transaction)

  Functions utilized: Option9

  Arguments passed:

  100000709 as transaction ID



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 9

Delete sales record

Please enter transaction id*: 100000709

{'date': '2024-03-01', 'trans_id': '100000709', 'customer_id': '100201', 'category': 'alcohol', 'value': '1200'}
Transaction deleted
```

Conclusion: Feature meets requirement.

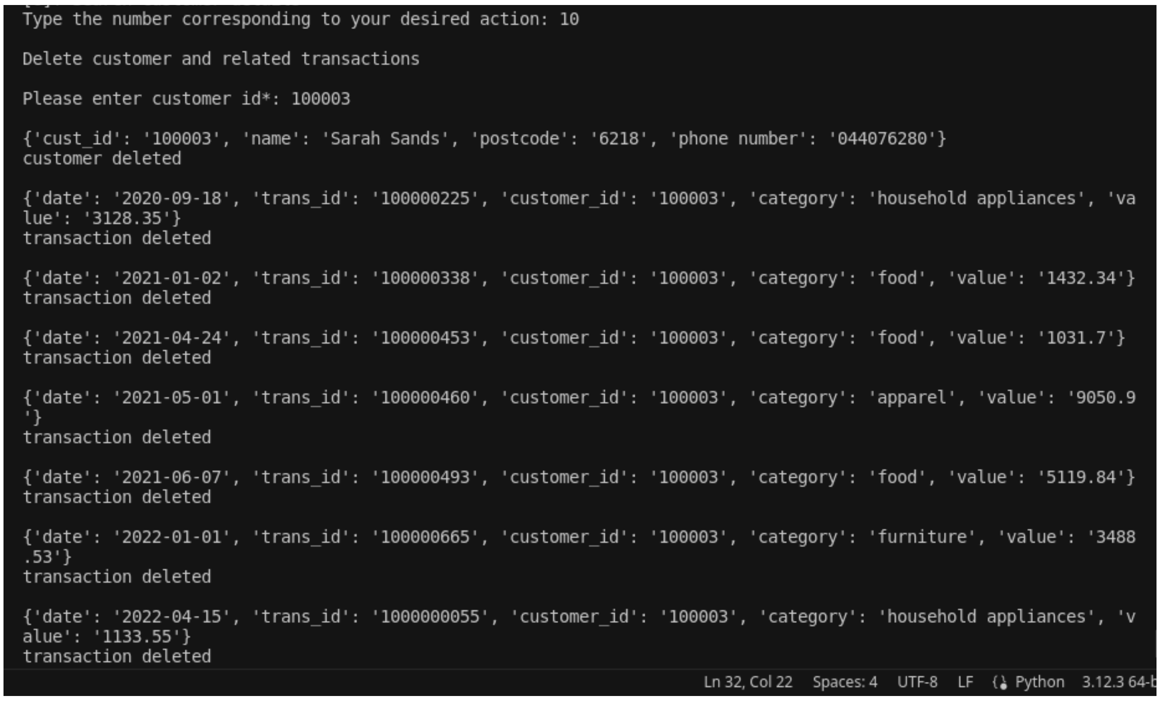
- Feature: Delete a customer and all related sales details

  Test Case: customer exists in data with some transactions for Option 10 (deleting a customer and related transactions)

  Functions utilized: Option10

  Arguments passed:

  100003 as customer ID



```
Type the number corresponding to your desired action: 10

Delete customer and related transactions

Please enter customer id*: 100003

{'cust_id': '100003', 'name': 'Sarah Sands', 'postcode': '6218', 'phone number': '044076280'}
customer deleted

{'date': '2020-09-18', 'trans_id': '100000225', 'customer_id': '100003', 'category': 'household appliances', 'va
lue': '3128.35'}
transaction deleted

{'date': '2021-01-02', 'trans_id': '100000338', 'customer_id': '100003', 'category': 'food', 'value': '1432.34'}
transaction deleted

{'date': '2021-04-24', 'trans_id': '100000453', 'customer_id': '100003', 'category': 'food', 'value': '1031.7'}
transaction deleted

{'date': '2021-05-01', 'trans_id': '100000460', 'customer_id': '100003', 'category': 'apparel', 'value': '9050.9
'}
transaction deleted

{'date': '2021-06-07', 'trans_id': '100000493', 'customer_id': '100003', 'category': 'food', 'value': '5119.84'}
transaction deleted

{'date': '2022-01-01', 'trans_id': '100000665', 'customer_id': '100003', 'category': 'furniture', 'value': '3488
.53'}
transaction deleted

{'date': '2022-04-15', 'trans_id': '1000000055', 'customer_id': '100003', 'category': 'household appliances', 'v
alue': '1133.55'}
transaction deleted
                                        Ln 32, Col 22   Spaces: 4   UTF-8   LF   Python   3.12.3 64-b
```

Conclusion: Feature meets requirement.

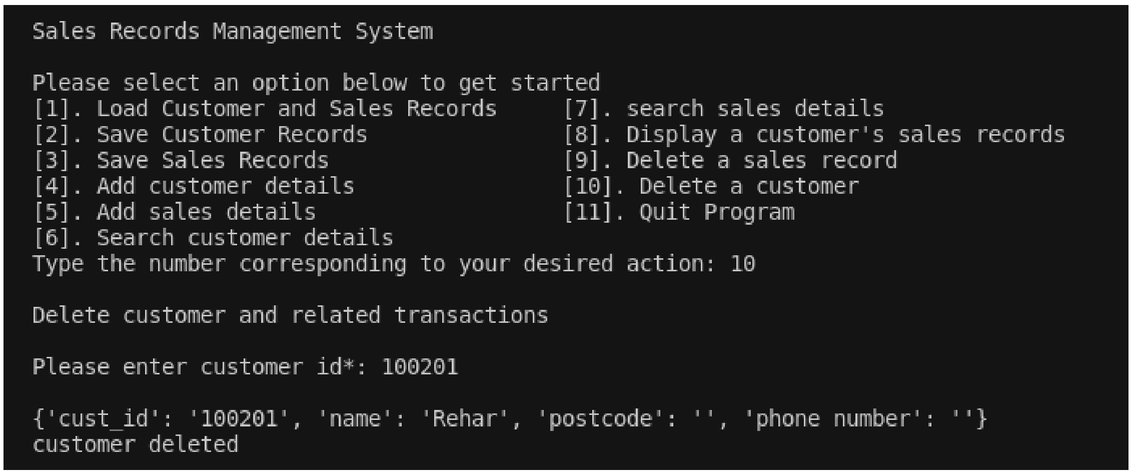
- Feature: Delete a customer and all related sales details

  Test Case: customer exists in data with no transactions for Option 10 (deleting a customer and related transactions)

  Functions utilized: Option10

  Arguments passed:

  100201 as customer ID



```
Sales Records Management System

Please select an option below to get started
[1]. Load Customer and Sales Records     [7]. search sales details
[2]. Save Customer Records               [8]. Display a customer's sales records
[3]. Save Sales Records                  [9]. Delete a sales record
[4]. Add customer details                [10]. Delete a customer
[5]. Add sales details                   [11]. Quit Program
[6]. Search customer details
Type the number corresponding to your desired action: 10

Delete customer and related transactions

Please enter customer id*: 100201

{'cust_id': '100201', 'name': 'Rehar', 'postcode': '', 'phone number': ''}
customer deleted
```

Conclusion: Feature meets requirement.

D. Filenames:

- question_2.py: Contains the main program logic and menu handling.
- functions_1.py: Contains the functions from question 1
- functions_2.py: Contains Menu and all functions for question 2
- Q2 test data (Dir):
  - customers.csv: Example CSV file for customer records.
  - sales.csv: Example CSV file for sales records.
- Unit tests (Dir): Contains unit tests with sample data.

IV. Question 3: Display sales performance graphically

A. Discussion of the Solution

The Sales Records Management System efficiently manages and visualizes sales data through its menu-driven interface. Integration of customer and postcode-specific sales records and dynamic y-tick adjustment enables clear and intuitive visualization. Using dual y-axes to display monthly sales value and count simultaneously enhances user insights, drawing inspiration from relevant tutorials. Despite these improvements, the system lacks robust error handling, refactoring opportunities, and comprehensive documentation, hindering usability and maintainability.

The system's strengths include its capability to translate raw sales data into clear visual representations and offer intuitive insights into sales patterns. Dynamic allocation of y-axis tick sizes improves readability and interpretability, bolstering the system's effectiveness in conveying information. Furthermore, integrating numpy and matplotlib ensures efficient data processing and flexible visualization options, supporting usability and adaptability.

However, potential weaknesses arise from scalability issues with large datasets and the absence of a graphical user interface, potentially impacting user experience. To mitigate these drawbacks, optimizing data processing for larger datasets and introducing a graphical user interface could be beneficial. Additionally, despite efforts to align the grid between y-axes for enhanced interpretability, challenges were encountered, resulting in its removal. The inclusion of the grid could have made plots more easily interpretable. Overall, error handling, interface design, and performance optimization enhancements are crucial for improving the system's reliability and usability in managing and visualizing sales records.

B. Self-Diagnosis and Evaluation

Features Completed and Working

- Option 11: Successfully plots monthly sales data using a line graph.
- Option 12: Generates a dual-axis plot showing both monthly sales and the number of sales for a specific customer.
- Option 13: Plots monthly sales and the number of sales for customers in a specified postcode.

Features Completed but Not Fully Working

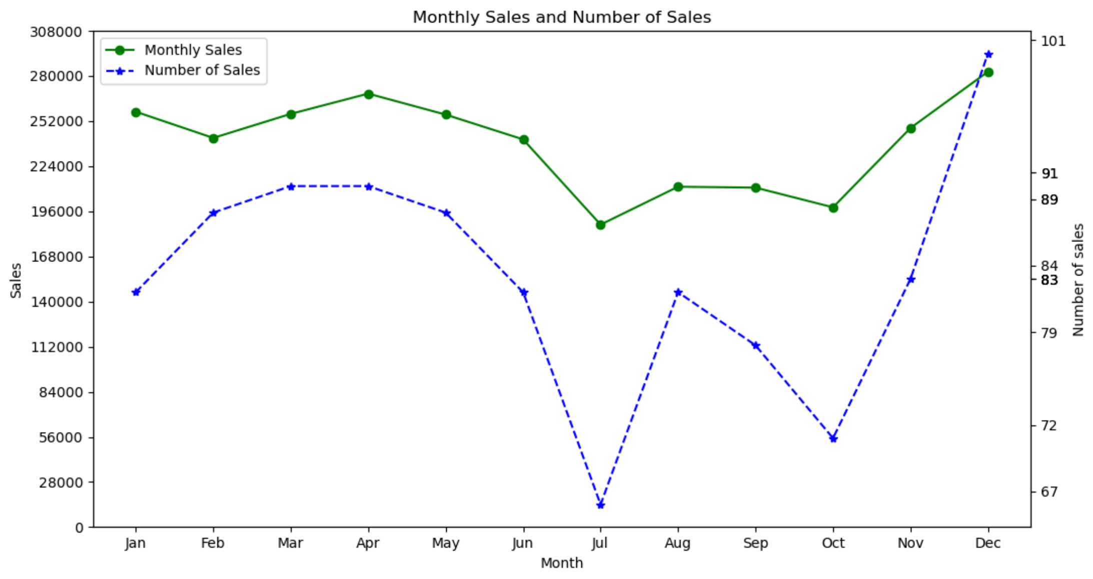
- Step Size Calculation: Initially, we had issues with zero step sizes, but this has been addressed. Further refinement might be needed for edge cases.

Features Not Fully Completed or Not Attempted

- Advanced Error Handling: While basic error handling is in place, more comprehensive checks and balances could be implemented.
- GUI Development: No graphical user interface has been attempted, which could significantly enhance usability.

C. Test Evidence

- Feature: Display value of sales and number of sales per month
  Test Case: There is data in memory Option 11 (visualizing value of sales and number of sales per month)
  Functions utilized: Option11
  Arguments passed:
      None



Conclusion: Feature meets requirement.

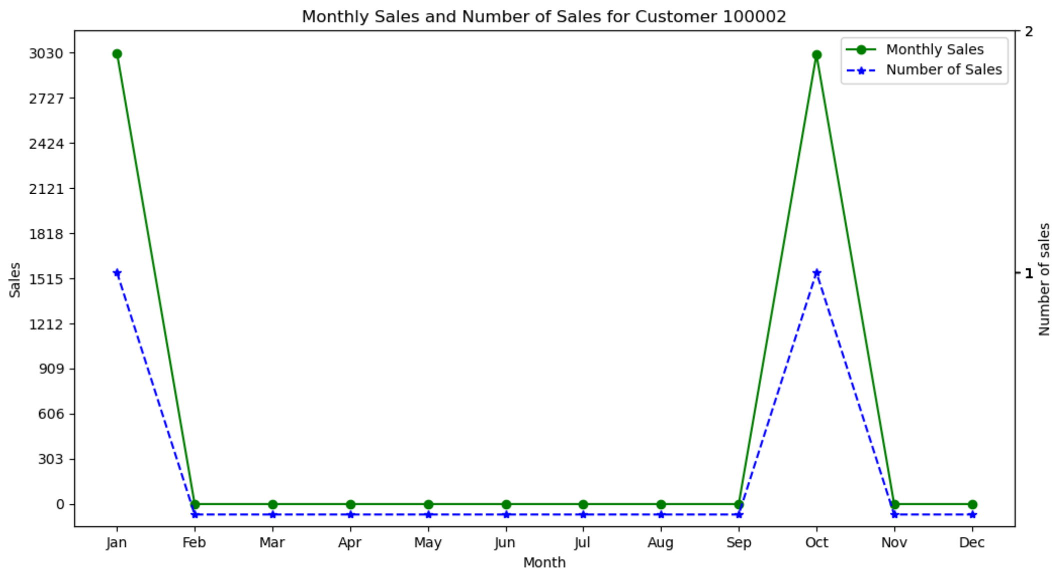
- Feature: Display the value of sales and number of sales per month for a given customer ID

  Test Case: the customer has little data for Option 12 (visualizing the value of sales and number of sales per month for a customer)

  Functions utilized: Option12

  Arguments passed:

  100002 as customer ID



Conclusion: Feature meets requirement.

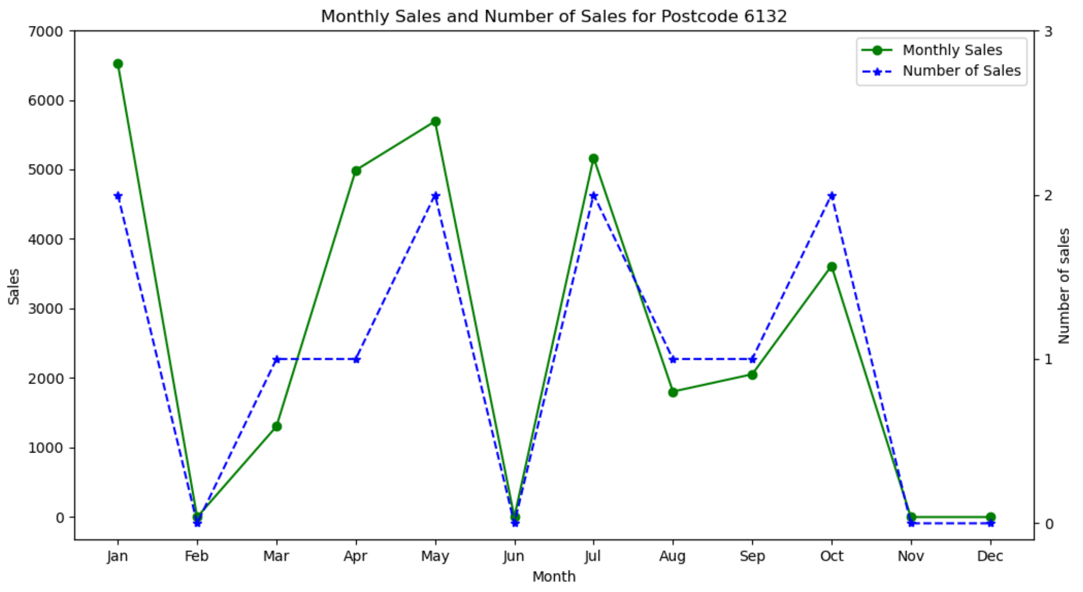
- Feature: Display the value of sales and number of sales per month for all customers in a postcode

  Test Case: There are customers in the postcode for Option 13 (visualizing value of sales and number of sales per month for all customers in a postcode)

  Functions utilized: Option13

  Arguments passed:

  6132 as postcode



Conclusion: Feature meets requirement.

D. Filenames:

- question_3.py: Contains the main program logic and menu handling.
- functions_1.py: Contains the functions from question 1
- functions_2.py: Contains the functions from question 2
- functions_3.py: Contains the menu and functions for features in question 3 .
- Q3 test data (Dir):
  - customers.csv: Example CSV file for customer records.
  - sales.csv: Example CSV file for sales records.
- Unit tests (Dir): Contains unit tests with sample data.

V. Conclusion:

In conclusion, all the features across the three questions have been coded and tested extensively in this project. Though the features have been built very well, we should work on the future considerations below to improve the system further.

VI. Future Work

To propel this project towards even greater heights, future endeavors could focus on several key areas of improvement:

1. **Graphical User Interface (GUI) Development:** Transitioning from a command-line interface to a user-friendly GUI could significantly enhance the system's accessibility and appeal to a broader user base. Implementing intuitive visual elements and interactive features would streamline user interaction and improve overall user experience.
2. **Scalability and Performance Optimization:** As datasets grow in size and complexity, optimizing the system's performance to handle large volumes of data efficiently becomes paramount. Exploring techniques such as database integration could mitigate performance bottlenecks and ensure seamless operation even under heavy workloads.
3. **Enhanced Error Handling and Documentation:** Strengthening the system's error handling mechanisms and providing comprehensive documentation would enhance system reliability and facilitate easier troubleshooting. Clearer error messages and informative prompts would empower users to confidently navigate the system, reducing the likelihood of errors and misunderstandings.
4. **Feature Expansion:** Continuously evolving business requirements may necessitate the addition of new features and functionalities to meet emerging needs. Future iterations of the Sales Records Management System could explore avenues such as predictive analytics, automated reporting, and machine learning integration to unlock deeper insights and drive informed decision-making.

By embracing these avenues for growth and innovation, the sales records management system could evolve into a properly Business-ready system.

Reference

Coderzcolumn. (n.d.). Secondary y-axis and x-axis matplotlib. Coderzcolumn. https://coderzcolumn.com/tutorials/data-science/secondary-y-axis-and-x-axis-matplotlib

Matplotlib. (n.d). https://matplotlib.org/

Numpy. 2024. https://numpy.org/doc/stable/reference/generated/numpy.zeros.html

W3Schools. (n.d). https://www.w3schools.com/