

# **K. R. MANGALAM UNIVERSITY**



## **ASSIGNMENT 2**

**Course Name: Programming for Problem Solving Using Python**

**Assignment Title: Contact Book**

**Student Name: Tushar Singh**

**Roll No.: 2501940055**

**Program: MCA (AI & ML)**

**Semester: I**

**Session: 2025-26**

**Faculty Name: Ms. Neha Kaushik**

**Date of Submission: 19/11/2025**

<b>Department:</b> SOET	<b>Session:</b> 2025-26
<b>Program:</b> MCA (AI/ML)	<b>Semester:</b> 1
<b>Course Code:</b> ETCCPP171	<b>College Roll no:</b> 2501940055
<b>Course Name:</b> Programming for Problem Solving Using Python	
<b>Submitted by:</b> Tushar Singh	<b>Faculty:</b> Ms. Neha Kaushik

**GitHub Link :-** <https://github.com/singhtushar738-prog/Contact-Book-Assignment.git>

**Name:** Tushar Singh

**Date:** 19/11/2025

**Assignment:** Contact Book

**Code:**

```

1 import csv
2 import json
3 import os
4 from datetime import datetime
5
6 CSV_PATH = "contact_list.csv"
7 JSON_PATH = "contact_list.json"
8 LOG_PATH = "error_log.txt"
9 CSV_COLUMNS = ["name", "phone", "email"]
10
11 def log_err(message, operation):
12     """Append an error message with timestamp and operation to the log file."""
13     try:
14         with open(LOG_PATH, "a", encoding="utf-8") as log:
15             timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
16             log.write(f"[{timestamp}] {operation}: {message}\n")
17     except Exception:
18         # if logging fails we silently ignore to avoid crashing the program
19         pass
20
21 def ensure_csv():
22     """Create CSV file with header row if it doesn't exist."""
23     if not os.path.exists(CSV_PATH):
24         try:
25             with open(CSV_PATH, "w", newline="", encoding="utf-8") as f:
26                 writer = csv.DictWriter(f, fieldnames=CSV_COLUMNS)
27                 writer.writeheader()
28         except Exception as e:
29             log_err(str(e), "Ensure CSV")
30
31 def load_contacts():
32     """Return list of contact_list as dicts. If file missing or empty returns
33     empty list."""
34     contact_list = []
35     try:
36         with open(CSV_PATH, "r", newline="", encoding="utf-8") as f:
37             reader = csv.DictReader(f)
38             for row in reader:
39                 # normalize keys and ensure fields exist
40
41                 contact_list.append({
42                     "name": (row.get("name") or "").strip(),
43                     "phone": (row.get("phone") or "").strip(),
44                     "email": (row.get("email") or "").strip(),

```

```

45         })
46     except FileNotFoundError:
47         # handled by caller or will be created
48         pass
49     except Exception as e:
50         log_err(str(e), "Read Contacts")
51     return contact_list
52 def save_contacts(contact_list):
53     """Overwrite CSV with contact_list (list of dicts)."""
54     try:
55         with open(CSV_PATH, "w", newline="", encoding="utf-8") as f:
56             writer = csv.DictWriter(f, fieldnames=CSV_COLUMNS)
57             writer.writeheader()
58             for c in contact_list:
59                 writer.writerow({k: c.get(k, "") for k in CSV_COLUMNS})
60     except Exception as e:
61         log_err(str(e), "Write Contacts")
62         raise
63 def create_contact():
64     try:
65         name = input("Enter Name: ").strip()
66         if not name:
67             print("Name cannot be empty. Aborting add.\n")
68             return
69         phone = input("Enter Phone Number: ").strip()
70         email = input("Enter Email Address: ").strip()
71         contact_list = load_contacts()
72
73         # Prevent exact duplicate by name+phone
74         for c in contact_list:
75             if c["name"].lower() == name.lower() and c["phone"] == phone:
76                 print("A contact with the same name and phone already exists.\n")
77                 return
78         contact_list.append({"name": name, "phone": phone, "email": email})
79         save_contacts(contact_list)
80         print("Contact added successfully.\n")
81
82     except Exception as e:
83         log_err(str(e), "Add Contact")
84         print("Error while adding contact.\n")
85
86 def show_contacts():
87     try:
88         contact_list = load_contacts()
89         if not contact_list:
90             print("No contact_list found.\n")
91             return
92
93         # nice table-like printing with index
94         print("\nIndex | Name | Phone | Email")
95         print("-" * 70)
96         for i, c in enumerate(contact_list, start=1):
97             print(f"{i:5d} | {c['name'][:24]:24} | {c['phone'][:15]:15} | {c['email'][:15]:15}")
98         print()
99
100     except Exception as e:
101         log_err(str(e), "Display Contacts")
102         print("Error displaying contact_list.\n")
103
104 def find_contact():

```

```

107     query = input("Enter name or phone to search (partial allowed):
108 ").strip().lower()
109     if not query:
110         print("Empty query.\n")
111         return
112
113     try:
114         contact_list = load_contacts()
115         results = [c for c in contact_list if query in c["name"].lower() or query
116 in c["phone"].lower()]
117
118         if results:
119             print(f"\nFound {len(results)} matching contact(s):")
120             print("-" * 50)
121             for c in results:
122                 print(f"Name: {c['name']} | Phone: {c['phone']} | Email:
123 {c['email']}")
124             print()
125         else:
126             print("Contact not found.\n")
127
128     except Exception as e:
129         log_err(str(e), "Search Contact")
130         print("Error searching contact.\n")
131
132 def modify_contact():
133     try:
134         contact_list = load_contacts()
135         if not contact_list:
136             print("No contact_list to update.\n")
137             return
138
139         show_contacts()
140         choice = input("Enter the Index of the contact to update (or name):
141 ").strip()
142
143         # find contact by index or name
144         contact = None
145         if choice.isdigit():
146             idx = int(choice) - 1
147             if 0 <= idx < len(contact_list):
148                 contact = contact_list[idx]
149         else:
150             for c in contact_list:
151                 if c["name"].lower() == choice.lower():
152                     contact = c
153                     break
154
155         if not contact:
156             print("Contact not found.\n")
157             return
158
159         print("Selected:")
160         print(f"Name: {contact['name']} | Phone: {contact['phone']} | Email:
161 {contact['email']}")
162         print("What do you want to update?")
163         print("1. Name")
164         print("2. Phone")
165         print("3. Email")
166         print("4. Cancel")
167         opt = input("Enter choice: ").strip()
168         if opt == "1":

```

```

169         new = input("Enter new name: ").strip()
170         if new:
171             contact["name"] = new
172     elif opt == "2":
173         new = input("Enter new phone: ").strip()
174         if new:
175             contact["phone"] = new
176     elif opt == "3":
177         new = input("Enter new email: ").strip()
178         if new:
179             contact["email"] = new
180     else:
181         print("Update cancelled.\n")
182         return
183
184     save_contacts(contact_list)
185     print("Contact updated successfully.\n")
186 except Exception as e:
187     log_err(str(e), "Update Contact")
188     print("Error updating contact.\n")
189
190 def remove_contact():
191     try:
192         contact_list = load_contacts()
193         if not contact_list:
194             print("No contact_list to delete.\n")
195             return
196         show_contacts()
197         choice = input("Enter the Index of the contact to delete (or exact name):
198 ").strip()
199
200         # find index
201
202         del_idx = None
203         if choice.isdigit():
204             idx = int(choice) - 1
205             if 0 <= idx < len(contact_list):
206                 del_idx = idx
207         else:
208             for i, c in enumerate(contact_list):
209                 if c["name"].lower() == choice.lower():
210                     del_idx = i
211                     break
212
213         if del_idx is None:
214             print("Contact not found.\n")
215             return
216         confirm = input(f"Are you sure you want to delete
217 '{contact_list[del_idx]['name']}'? (y/N): ").strip().lower()
218         if confirm != "y":
219             print("Delete cancelled.\n")
220             return
221         removed_contact = contact_list.pop(del_idx)
222         save_contacts(contact_list)
223         print(f"Deleted contact: {removed_contact['name']}\n")
224
225     except Exception as e:
226         log_err(str(e), "Delete Contact")
227         print("Error deleting contact.\n")
228
229 def export_json():
230     try:

```

```

231     contact_list = load_contacts()
232     with open(JSON_PATH, "w", encoding="utf-8") as file:
233         json.dump(contact_list, file, indent=4, ensure_ascii=False)
234     print("Contacts exported to JSON successfully.\n")
235 except Exception as e:
236     log_err(str(e), "Export JSON")
237     print("Error exporting to JSON.\n")
238
239 def import_json():
240     try:
241         if not os.path.exists(JSON_PATH):
242             print("JSON file not found.\n")
243             return
244         with open(JSON_PATH, "r", encoding="utf-8") as file:
245             contact_list = json.load(file)
246
247         if not contact_list:
248             print("No contact_list in JSON file.\n")
249             return
250
251         print("\nContacts from JSON:")
252         print("-" * 50)
253         for c in contact_list:
254             print(f"Name: {c.get('name', '')} | Phone: {c.get('phone', '')} | Email:
255 {c.get('email', '')}")
256         print()
257
258     except Exception as e:
259         log_err(str(e), "Load JSON")
260         print("Error loading JSON.\n")
261
262 def menu_main():
263     ensure_csv()
264     print("Welcome to the Contact Book Manager!")
265
266     while True:
267         print("1. Add Contact")
268         print("2. View Contacts")
269         print("3. Search Contact")
270         print("4. Update Contact")
271         print("5. Delete Contact")
272         print("6. Export to JSON")
273         print("7. Load from JSON")
274         print("8. Exit")
275
276     choice = input("Enter your choice: ").strip()
277
278     if choice == "1":
279         create_contact()
280     elif choice == "2":
281         show_contacts()
282     elif choice == "3":
283         find_contact()
284     elif choice == "4":
285         modify_contact()
286     elif choice == "5":
287         remove_contact()
288     elif choice == "6":
289         export_json()
290     elif choice == "7":
291         import_json()
292     elif choice == "8":

```

```

293         print("Exiting program.")
294         break
295     else:
296         print("Invalid choice. Try again.\n")
297
298 if __name__ == "__main__":
299     menu_main()

```

## Output:

```

Welcome to the Contact Book Manager!
1. Add Contact
2. View Contacts
3. Search Contact
4. Update Contact
5. Delete Contact
6. Export to JSON
7. Load from JSON
8. Exit
Enter your choice: 1
Enter Name: Tushar Sharma
Enter Phone Number: 9941545
Enter Email Address: hsbdyfbsegj@gmail.com
Contact added successfully.

1. Add Contact
2. View Contacts
3. Search Contact
4. Update Contact
5. Delete Contact
6. Export to JSON
7. Load from JSON
8. Exit
Enter your choice: 7
JSON file not found.

1. Add Contact
2. View Contacts
3. Search Contact
4. Update Contact
5. Delete Contact
6. Export to JSON
7. Load from JSON
8. Exit
Enter your choice: 6
Contacts exported to JSON successfully.

1. Add Contact
2. View Contacts
3. Search Contact
4. Update Contact
5. Delete Contact
6. Export to JSON
7. Load from JSON
8. Exit
Enter your choice: 7

Contacts from JSON:
-----
Name: Tushar Sharma | Phone: 9941545 | Email: hsbdyfbsegj@gmail.com

```

Enter your choice: 1  
Enter Name: Vinayak Sharma  
Enter Phone Number: 9595952141  
Enter Email Address: vinayakshar@gmail.com  
Contact added successfully.

1. Add Contact  
2. View Contacts  
3. Search Contact  
4. Update Contact  
5. Delete Contact  
6. Export to JSON  
7. Load from JSON  
8. Exit  
Enter your choice: 7

Contacts from JSON:  
-----  
Name: Tushar Sharma | Phone: 9941545 | Email: hsbdyfbsegj@gmail.com

1. Add Contact  
2. View Contacts  
3. Search Contact  
4. Update Contact  
5. Delete Contact  
6. Export to JSON  
7. Load from JSON  
8. Exit  
Enter your choice: 6  
Contacts exported to JSON successfully.

1. Add Contact  
2. View Contacts  
3. Search Contact  
4. Update Contact  
5. Delete Contact  
6. Export to JSON  
7. Load from JSON  
8. Exit  
Enter your choice: 7

Contacts from JSON:  
-----  
Name: Tushar Sharma | Phone: 9941545 | Email: hsbdyfbsegj@gmail.com  
Name: Vinayak Sharma | Phone: 9595952141 | Email: vinayakshar@gmail.com

1. Add Contact  
2. View Contacts  
3. Search Contact  
4. Update Contact  
5. Delete Contact  
6. Export to JSON  
7. Load from JSON  
8. Exit  
Enter your choice: 8