**A PROJECT REPORT**

**ON**

**Contact Management System Using Client Server Model**

**Submitted by**

**Name of the candidates**

1. **Devendra Mahor**
2. **Ujwala Chounde**

**Guided by**

**Name of the mentors**

**Prachi Mittal**

**Capgemini Technology Services India Limited**

****

**ABSTRACT**

**Project in C Contact Management System** is a simple console application without graphics. It is similar to the contact manager in cell phones. In this project, you can add, view, edit, search and delete contacts. All added and edited records are saved in a file.

You can list contacts by name, phone no., address and email. File handling has been used to record all data. I have used data structure to store the user name, email and contact. Overall, understanding the simple source code of this project will teach you how to add, edit, search, list and remove data using file.

**CONTENTS**

1. Introduction……………………………………………………4
2. Data Structures & Search Methods Used……………………..5
3. File Structure Used……………………………………………5
4. Client Server Model…………………………………………..6
5. Unit Test Plan & Results……………………………………...7
6. Static Analysis Reports………………………………………..8
7. Valgrind Report for Memory Leaks………………………….9

**Introduction**

About Contact Management System in C:

File handling, data structure, functions, and pointers are the main things which make up this simple[C project](https://www.codewithc.com/tag/c-mini-projects). The key features of contact management system are listed below:

* Add new contacts: with information such as name, phone number, address, and email
* List all contacts: lists all the contacts stored in file with their respective contact details
* Search contacts: based on name and phone number
* Edit contacts: edit information given while adding the contacts – name, phone number, address, and email
* Delete contacts: deletes contacts from file
* Handling multiple client requests by semaphores.

**Data Structure**

In this project, we have used user- defined data structure. Structure is a user-defined datatype in C language which allows us to combine data of different types together.

**Searching Methods**

Searching is the process of finding some particular element in the list. If the element is present in the list, then the process is called successful, and the process returns the location of that element; otherwise, the search is called unsuccessful. In this project, we have used Linear searching. Linear search is also called as **sequential search algorithm.** It is the simplest searching algorithm. In Linear search, we simply traverse the list completely and match each element of the list with the item whose location is to be found. If the match is found, then the location of the item is returned; otherwise, the algorithm returns NULL.

**File Structure**A structure of (name,email,contact\_no,address,func\_name),

Stored in file format.

**Client Server Model**

In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client. Clients do not share any of their resources.

**How the Client-Server Model works?**

* **Client:** When we talk the word **Client**, it means to talk of a person or an organization using a particular service. Similarly, in the digital world a **Client** is a computer (**Host**) i.e. capable of receiving information or using a particular service from the service providers (**Servers**).
* **Servers:** Similarly, when we talk the word **Servers**, it means a person or medium that serves something. Similarly, in this digital world a **Server** is a remote computer which provides information (data) or access to particular services.

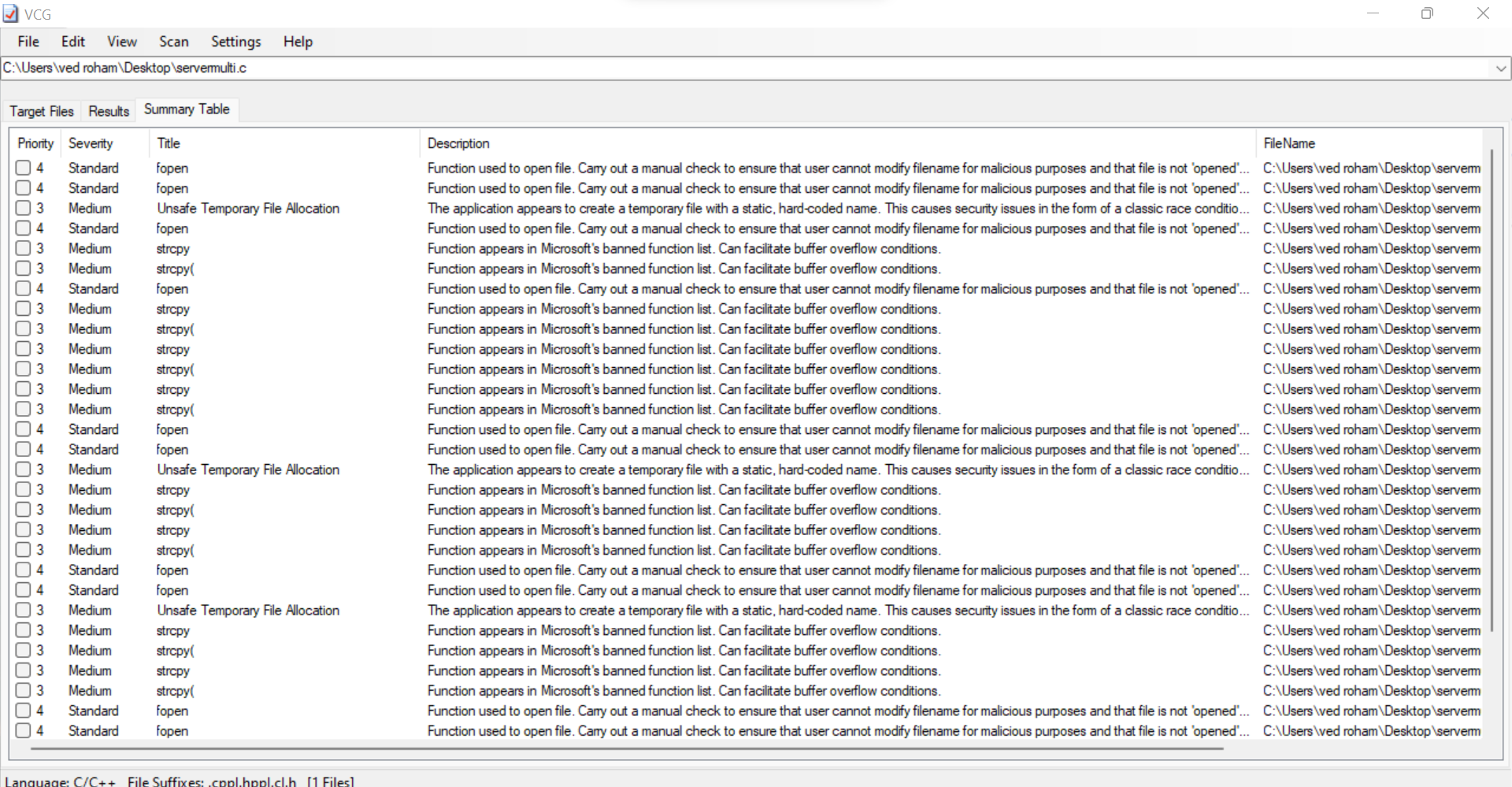
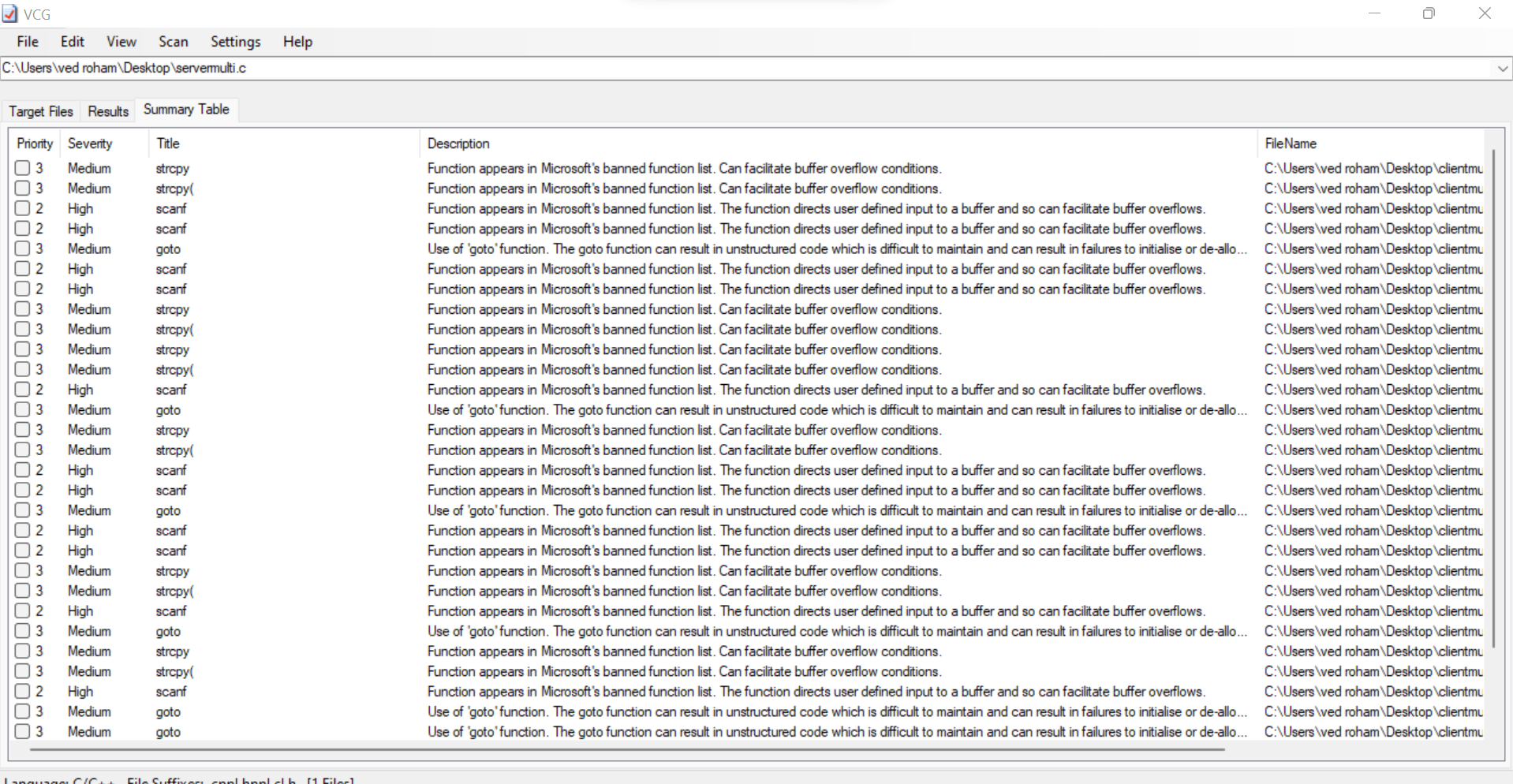
So, it’s basically the **Client** requesting something and the **Server** serving it as long as its present in the database.



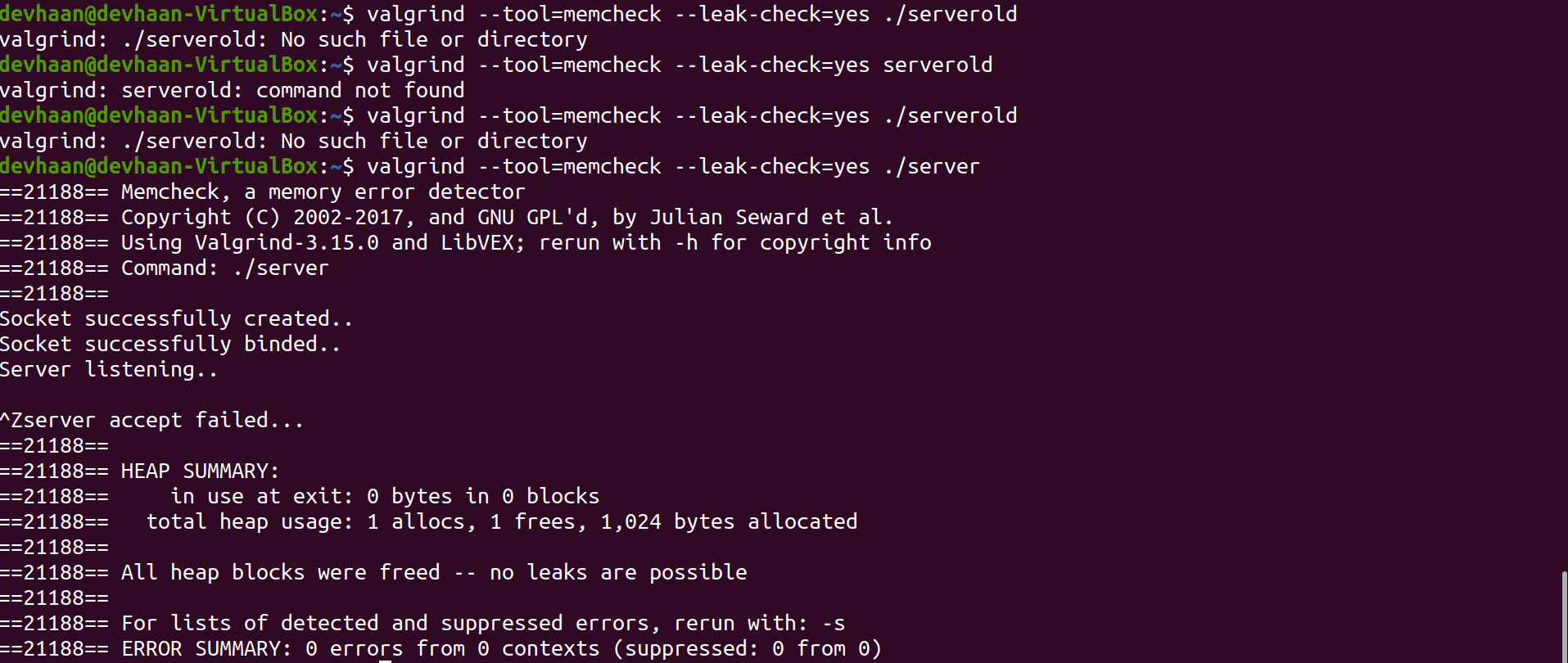
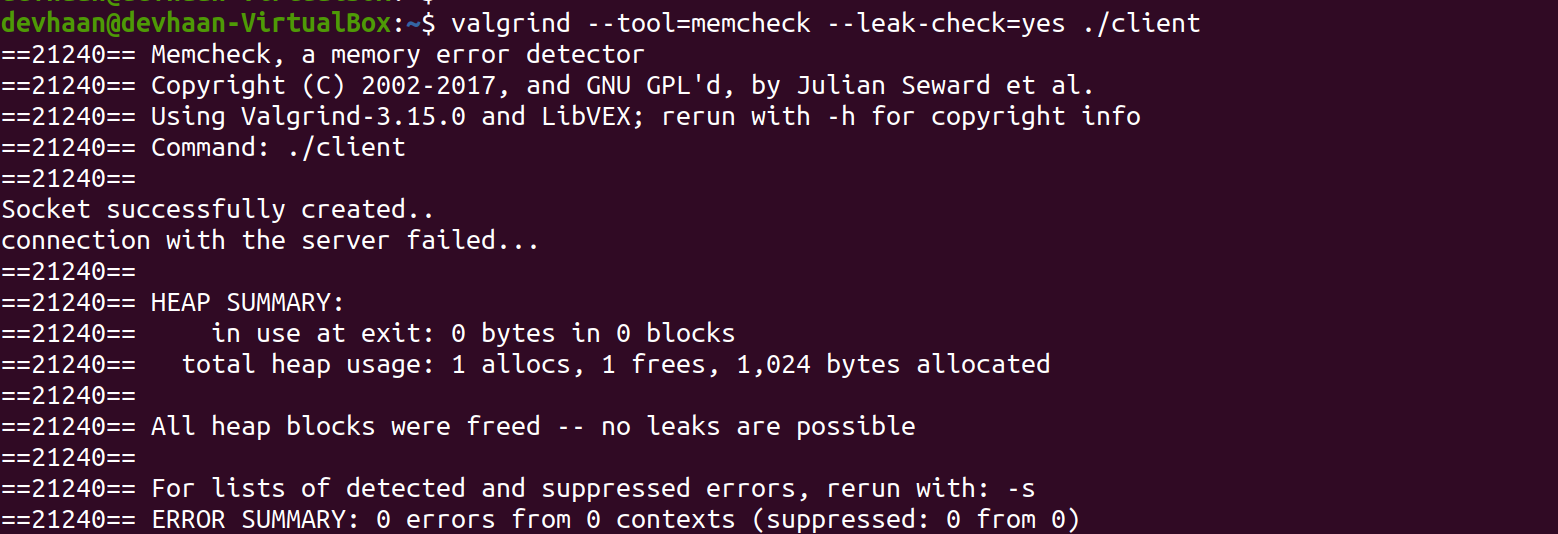
**Unit Test Plan & Results**

|  |  |  |
| --- | --- | --- |
| **NUMBER** | **Test Cases** | **Result** |
| 1 | Server started | Pass |
| 2 | Client-1 connected | Pass |
| 3 | Client-2 connected | Pass |
| 4 | Client-3 connected | Pass |
| 5 | Client-2 disconnected | Pass |
| 6 | Add contact | Pass |
| 7 | Delete contact | Pass |
| 8 | Edit contact | Pass |
| 9 | Search contact | Pass |
| 10 | Exit contact | Pass |
| 11 | Duplicate contact add | Pass |
| 12 | Shared data secure | Pass |
| 13 | Multiple clients connected | Pass |
| 14 | Delete non existed data | pass |
| 15 | Delete existed contact | pass |
| 16 | Edit existed contact | pass |
| 17 | Edit non existed contact | pass |
| 18 | Add invalid number | pass |
| 19 | Add already existed number | pass |
| 20 | Search non existed number | pass |
| 21 | Search valid data | pass |
| 22 | Delete contact via key value | pass |
| 23 | Add contact acknowledge | pass |

**Static Analysis Reports**

****

**Valgrind Report for Memory Leaks**

****