

CSE-2003

J-COMPONENT REPORT



November 2, 2020



VIT UNIVERSITY, CHENNAI

Vandalur – Kelambakkam Road

Chennai – 600127

***“SMART CONTACT BOOK”***

by

Shushant Singh (19BCE1620)

Apoorv Yadav (19BCE1163)

Aarohan B (19BCE1601)

Divyanu Baheti(19BCE1045)

A project report submitted to

**Dr. PATTABIRAMAN V**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

## CERTIFICATE

Certified that this project report entitled “***Smart Contact Book*”** is a bonafide work of Divyanu, Apoorv, Shushant and Aarohan who carried out the “J”-Project work under my supervision and guidance for CSE-2003.

**Dr. Pattabiraman V**

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING VIT University, Chennai

Chennai – 600 127.

## ABSTRACT

We have a dataset of year 2014 of a company named “Capital Bikeshare” which contains information about different bikes along with their distinct numbers that are being used by either a member or a casual user for their journey from one station to another provided with the start time and end time.

Through this project we will be designing a new way of accessing contacts or say a “Smart Contact Book”.

We will be making an efficient way of saving contacts with taking in consideration for our users to have an easy access to their contacts. We made a visually interactive output for better consumption of data.

## ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project faculty, **Dr. Pattabiraman V,** for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

Finally, we would like to thank our deemed university, VIT Chennai, for providing us with the opportunity and facilities which ensured this project’s completion.

## INTRODUCTION

We have a dataset of year 2014 of a company named “Capital Bikeshare” which contains information about different bikes along with their distinct numbers that are being used by either a member or a casual user for their journey from one station to another provided with the start time and end time.

We will be using File handling and Linked List.

In computer science, a **linked list** is a linear collection of data elements whose order is not given by their physical placement in memory. Instead, each element points to the next. It is a data structure consisting of a collection of nodes which together represent a sequence.

**File Handling** is the storing of data in a **file** using a program. In C programming language, the programs store results, and other data of the program to a **file** using **file handling** in C. Also, we can extract/fetch data from a **file** to work with it in the program.

1. **DATASET USED**

The project contains 2 datasets

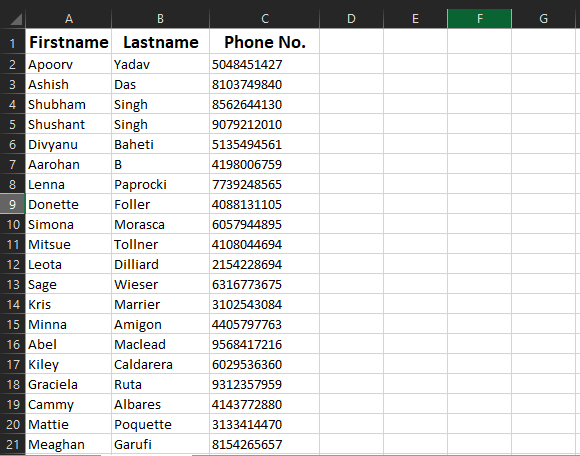
The first one contains the main contacts. The file contains the following headings:

* First name
* Last name
* Phone number

➢ This dataset comes from a site Letter Hub.

➢ There are a total of 152 contacts in the dataset.

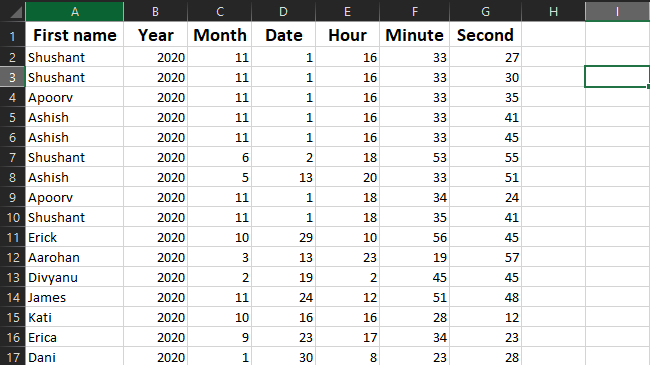
The Screenshot of the csv file:



Another csv is the call logs file that is generated after every calling. It contains the headings:

* First name
* Year
* Month
* Date
* Hour
* Minute
* Seconds

The screen shot of the file:



1. **IMPLEMENTATION CODE**
2. # include<stdio.h>
3. # include<conio.h>
4. # include<string.h>
5. # include<stdlib.h>
6. # include<time.h>
7. *void* menu();
8. *void* adddetails();
9. *void* searchdetails();
10. *void* modifydetails();
11. *void* deletedetails();
12. *void* display();
13. *void* call();
14. *void* smartsortfunc();
15. *void* load\_data();
16. *void* update\_file();
17. *void* sort();
18. *void* old\_record();
19. *struct* contact {
20. *char* lastname[20], firstname[20], mob\_no[15];//Defining our Linked List//
21. *int* count;
22. *struct* contact \* next;
23. };
24. typedef *struct* contact node;
25. node \* head, \* newnode;
26. *void* load\_data()
27. {
28. FILE \*fp;
29. fp = fopen("contact\_list.csv","r");
30. if(fp == NULL)
31. {
32. printf("contact\_list.csv file failed to open");
33. exit(1);
34. }
35. *char* line[200];
36. while(fgets(line,sizeof(line), fp))
37. {
38. node \* ptr, \* previous, \* newnode;
39. newnode = (node \* ) malloc(sizeof(node));
40. *char* \*token;
41. token = strtok(line, ",");
42. for(*int* i=0;token!=NULL;i++)
43. {
44. if(i==0)
45. {
46. strcpy(newnode -> firstname, token);
47. }
48. if(i==1)
49. {
50. strcpy(newnode -> lastname, token);
51. }
52. if(i==2)
53. {
54. *int* len = strlen(token)-1;
55. token[len] = '\0';
56. strcpy(newnode -> mob\_no, token);
57. newnode->count=0;
58. }
59. token = strtok(NULL,",");
60. }
61. newnode -> next = NULL;
62. if (head == NULL) {
63. head = newnode;//If there is no linked list then add the details to the head//
64. } else {
65. previous = ptr = head;
66. //This loop sorts out the Contact in ascending order.//
67. while (strcmp(newnode -> firstname, ptr -> firstname) > 0) {
68. previous = ptr;
69. ptr = ptr -> next; //The loops breaks if it is at the end of the linked list or //
70. if (ptr == NULL) break;//If the character of the new node is less than the character of the new node ascii.//
71. }
72. if (ptr == previous) {
73. newnode -> next = head;//Insert at the 1st positon.//
74. head = newnode;
75. } else if (ptr == NULL) {
76. previous -> next = newnode;//Insert in the last position//
77. } else {
78. newnode -> next = ptr;
79. previous -> next = newnode;//Insert in the middle//
80. }
81. }
82. }
83. fclose(fp);
84. }
85. *void* old\_record()
86. {
87. FILE \*fp,\*np;
88. fp=fopen("call\_log.csv","r");
89. np=fopen("call\_log1.csv","wb+");
90. *time\_t* t= time(NULL);
91. *struct* tm tm = \*localtime(&t);
92. *int* cur\_year = tm.tm\_year+1900;
93. *int* cur\_month = tm.tm\_mon + 1;
94. if(fp==NULL)
95. {
96. printf("Failed to open file call\_log.csv");
97. exit(1);
98. }
99. *char* line[200];
100. while(fgets(line,sizeof(line), fp))
101. {
102. *char* \*token;
103. *char* name[30];
104. *int* year,month,date,min,sec,r\_year,r\_month,hour;
105. token = strtok(line, ",");
106. for(*int* i=0; token!=NULL; i++)
107. {
108. if(i==0)
109. {
110. strcpy(name, token);
111. }
112. if(i==1)
113. {
114. r\_year=atoi(token);
115. }
116. if(i==2)
117. {
118. r\_month=atoi(token);
119. }
120. if(i==3)
121. {
122. date = atoi(token);
123. }
124. if(i==4)
125. {
126. hour=atoi(token);
127. }
128. if(i==5)
129. {
130. min=atoi(token);
131. }
132. if(i==6)
133. {
134. sec=atoi(token);
135. }
136. token = strtok(NULL,",");
137. }
138. year= r\_year;
139. month= r\_month;
140. if(cur\_year == year && month< cur\_month-6)
141. {
142. continue;
143. }
144. if(year < cur\_year){
145. month = month +6;
146. if(month > 12)
147. {
148. year =year +1;
149. month = month -12;
150. }
151. if(year<cur\_year || (cur\_year == year && month< cur\_month)){
152. continue;
153. }
154. }
155. fprintf(np,"%s,%d,%d,%d,%d,%d,%d\n",name,r\_year,r\_month,date,hour,min,sec);
156. }
157. fclose(fp);
158. fclose(np);
159. remove("call\_log.csv");
160. rename("call\_log1.csv","call\_log.csv");
161. }
162. *void* smartsortfunc()
163. {
164. *struct* data{
165. *char* name[30];
166. *int* count;
167. };
168. FILE \*fp;
169. fp = fopen("call\_log.csv","r");
170. *time\_t* t= time(NULL);
171. *struct* tm tm = \*localtime(&t);
172. *int* cur\_hour = tm.tm\_hour;
173. *int* cur\_min = tm.tm\_min;
174. if(fp == NULL)
175. {
176. printf("call\_log.csv file failed to open");
177. exit(1);
178. }
179. node \* ptr;
180. for (ptr = head; ptr != NULL; ptr = ptr -> next){
181. ptr->count=0;
182. }
183. *char* line[200];
184. while(fgets(line,sizeof(line), fp))
185. {
186. *char* \*token;
187. *char* name[30];
188. *int* hour,min;
189. token = strtok(line, ",");
190. for(*int* i=0;token!=NULL;i++)
191. {
192. if(i==0)
193. {
194. strcpy(name, token);
195. }
196. if(i==4)
197. {
198. hour=atoi(token);
199. }
200. if(i==5)
201. {
202. min=atoi(token);
203. }
204. token = strtok(NULL,",");
205. }
206. if(cur\_hour-1<=hour && hour<=cur\_hour+1){
207. if(hour==cur\_hour-1 && min<cur\_min)
208. {
209. continue;
210. }
211. if(hour==cur\_hour+1 && min>cur\_min)
212. {
213. continue;
214. }
215. node \* ptr;
216. ptr = head;
217. while (strcmp(ptr -> firstname, name) != 0) {
218. ptr = ptr -> next;
219. if (ptr == NULL) break;
220. }
221. if (ptr != NULL) {
222. ptr -> count++;
223. } else {
224. printf("No such record Found .......\n");
225. }
226. }
227. }
228. for (ptr = head; ptr != NULL; ptr = ptr -> next){
229. printf("\nThe count of %s is %d",ptr->firstname,ptr->count);
230. }
231. fclose(fp);
232. sort();
233. printf("\n\t\t------------------------------\n");
234. for (ptr = head; ptr != NULL; ptr = ptr -> next) {
235. printf("\t\tFirst name: %s", ptr -> firstname);
236. printf("\n\t\tLast name:%s", ptr -> lastname);
237. printf("\n\t\tTelephone No.: %s", ptr -> mob\_no);
238. printf("\n\t\t------------------------------\n");
239. }
240. update\_file();
241. load\_data();
242. getch();
243. menu();
244. }
245. *int* main() {
246. head = (node \* ) malloc(sizeof(node));
247. head = NULL;
248. load\_data();
249. old\_record();
250. menu();
251. return 0;
252. }

**CONCLUSION**

* By the end of our project u will be getting a brand-new contact book software with much faster working and efficiency.
* It will be very user-friendly and will be very easy to access.
* It will save those little time of our users which will in hindsight save a lot time.

## 

.