

# **JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY**

("Deemed to be University" under Section 3 of UGC Act 1956)

## **Btech Sem-1 PROJECT REPORT**

### **Parking Database Management System**



Batch- D1

Submitted to:

Dr. Niyati Aggrawal  
Dr. Megha Rathi

Submitted by:

Mahika Jain (2501270012)

Utsavi Rawat (2501270006)

Dishi (2501270013)

Govind Agarwal (2501270026)

# **TABLE OF CONTENTS**

- **Summary**
  - Brief overview of the project
  - Objectives
- **Introduction**
  - Background and Context
  - Problem Statement
  - Scope of the Project
- **System Requirements**
  - Functional and Non -Functional Requirements
- **Design and Implementation**
  - Flow Chart
  - Code snippets demonstrating the use of C concepts
  - Output screenshots with description
- **Conclusion**
  - Summary & achievement of Objectives
  - Future Work and Recommendations
- **References**
  - List of all sources cited in the report

## **SUMMARY**

### **Overview of the project:**

Managing parking in busy places like malls, hospitals, and office campuses can often be confusing and time-consuming. The Parking Database Management System aims to make this process simpler, faster, and more organized. This project uses the basic yet powerful concepts of the C programming language to create a system that can record and manage parking information efficiently.

The system offers two user-friendly options for customers: they can either interact with a staff member at the counter manually , or they can choose the Automatic Mode and access parking services independently using a QR code. In both cases, details such as entry time, type of vehicle, and available parking spots are handled automatically by the program. Customers receive a slip or message containing all the necessary information about their parking spot and charges.

When the customer exits, the system calculates the total amount to be paid based on the time the vehicle was parked, ensuring transparency and accuracy. The use of file handling also helps store and update data securely, making it easier for the management to keep track of records.

Overall, this project not only enhances the parking experience for users but also helps institutions operate more smoothly. It demonstrates how basic C programming concepts—like functions, loops, conditions, arrays, and pointers—can be combined to build a practical solution for a real-world problem.

## **Objectives:**

- To make the parking process quicker and more convenient for customers by reducing manual effort and waiting time.
- To accurately record important details such as entry time, exit time, vehicle type, and parking slot availability.
- To provide two easy ways for customers to use the system—either with the help of an operator or independently through a QR-based automatic mode.
- To automatically calculate parking charges based on the duration of the stay, ensuring fairness and transparency.
- To maintain organized and secure records of all parking activities using file handling for future reference and management needs.
- To apply and demonstrate fundamental concepts of the C programming language in a practical, real-world application.
- To create a user-friendly system that can be used in various public places like hospitals, malls, and corporate offices.

## INTRODUCTION

### **Background and Context:**

In today's fast-growing cities, parking has become a major concern in crowded public spaces like malls, hospitals, and office complexes. People often waste a lot of time searching for an empty parking spot, waiting in queues, or dealing with misplaced parking records. These issues not only create inconvenience for customers but also make it difficult for organizations to manage space efficiently and maintain accurate data.

As technology continues to improve everyday services, automation in parking systems has become an essential step toward smarter infrastructure. A computer-based parking management system can help reduce errors, avoid confusion, and provide a smooth flow of vehicles. It can also support staff by handling time-consuming tasks like keeping records, calculating charges, and checking spot availability.

This project, developed using the C programming language, aims to introduce a simple yet effective solution to these common parking problems. By combining digital record-keeping and automated features, the Parking Database Management System strives to make the parking experience more organized, transparent, and user-friendly for both customers and administrators.

## **Problem Statement:**

In many public places, managing parking manually often leads to confusion, longer waiting times, and mismanagement of available spaces. Without a proper system, it becomes difficult to track vehicle entries, check slot availability, and calculate parking charges accurately. This not only causes inconvenience to customers but also increases the workload on staff and can result in financial discrepancies for the organization.

To overcome these challenges, there is a need for a reliable and automated parking management system that can store and process data efficiently. The system should ensure smooth vehicle movement, quick access to information, and secure handling of parking records while minimizing human errors.

## **Scope of the Project:**

The Parking Database Management System focuses on improving the parking experience for both users and administrators by automating essential tasks. The system covers the entire process—from a vehicle's entry to exit—by recording details like vehicle type, time of entry, parking slot availability, and final payment.

It is designed to work in two modes: Manual Mode, where staff at the counter assist customers, and Automatic Mode, where users can independently access the system through a QR code. The project also includes features like generating a

parking slip or digital message for the customer and calculating charges based on the time spent in the parking area.

This system is suitable for various environments such as shopping malls, hospitals, and corporate offices. While the current version focuses on basic parking management using fundamental C programming concepts, it also lays the groundwork for future enhancements such as smart sensors and online slot booking. Overall, the project demonstrates how a simple software solution can help reduce errors, save time, and make parking operations more organized.

# **SYSTEM REQUIREMENTS**

## **Functional Requirements:**

The Parking Database Management System must perform the following functions to ensure smooth and efficient parking operations:

### **1. User Mode Selection**

The system should allow the customer to choose between Manual Mode and Automatic Mode at the time of entry.

### **2. Vehicle Information Entry**

It should record essential details such as the vehicle type (2-wheeler or 4-wheeler) and entry time.

### **3. Parking Slot Allocation**

The system must check for available parking spaces and assign a suitable slot based on the vehicle type.

### **4. Slip / Message Generation**

After allocating a parking spot, the system should generate a slip or a digital message showing the parking slot, check-in time, and reference charges.

## **5. Real-Time Updates**

Parking slot availability must update automatically when a vehicle arrives or exits.

## **6. Exit Processing**

At the time of exit, the system should record or detect the exit time and calculate the total parking duration.

## **7. Payment Calculation**

The system should compute the final amount based on the duration of the stay and display the amount payable.

## **8. Secure Payment Handling**

The system should allow customers to select their payment mode. (Cash preferred in Manual Mode; digital-only in Automatic Mode.)

## **9. Data Storage and Retrieval**

File handling must be used to store all parking records and update them whenever a vehicle enters or exits.

## **10. Smooth Navigation and Messaging**

The interface should guide the user through each step and confirm successful payment and exit.

## **Non -Functional Requirements:**

### **1.Usability**

The system should be simple and user-friendly so that both customers and staff can operate it without difficulty. Clear instructions and smooth navigation must be provided throughout the process.

### **2.Performance**

The system should respond quickly while checking slot availability, generating slips, and calculating charges, ensuring that customers do not face delays.

### **3.Accuracy**

All time records, slot updates, and payment calculations must be precise to avoid confusion or financial errors.

### **4.Reliability**

The system should consistently work without crashes or unexpected failures, especially during peak hours when parking activity is high.

### **5.Data Security**

The system must securely store parking records using file handling so that data is

not lost, misused, or tampered with.

## **6.Scalability**

It should be able to handle more vehicles and additional features in the future, allowing expansion to larger parking areas or multiple locations.

## **7.Maintainability**

The code should be well-structured so that future updates, fixes, or improvements can be made easily without affecting the existing functionality.

## **8.Compatibility**

The system should run smoothly on different computers or devices commonly used at parking facilities.

## DESIGN AND IMPLEMENTATION

Code snippets demonstrating the use of C concepts:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5
6 // Project - Parking Management System
7 // Table T1 -> 2 Wheeler vehicle
8 // Table T2 -> 4 Wheeler vehicles
9
10 //Table for Tower 1
11
12 typedef struct s1 {
13     int sno;
14     char date[20];
15     char tower_no[5];
16     int slot_no;
17     char status[20];
18     char carno[20];
19     char contact_no[20];
20     int enhour;
21     int enmin;
22 } s1;
23
24 //Table for Tower 2
25
26
27 typedef struct s2 {
28     int sno;
29     char date[20];
30     char tower_no[5];
31     int slot_no;
32     char status[20];
33     char carno[20];
34     char contact_no[20];
35     int enhour;
36     int enmin;
37 } s2;
38
39
40 void struct1(s1 T1[]);
41 void struct2(s2 T2[]);
42
43 // Functions definition
44
45 void save_T1(s1 T1[], int count) {
46     FILE *fp = fopen("dat1.txt", "w");
47     if (fp == NULL) {
48         printf("Error opening file \n");
49         return;
50     }
51 }
```

```

52     for (int i = 0; i < count; i++) {
53
54         fprintf(fp, "%d %s %s %d %s %s %s %d %d\n",
55                 T1[i].sno,
56                 T1[i].date,
57                 T1[i].tower_no,
58                 T1[i].slot_no,
59                 T1[i].status,
60                 T1[i].carno,
61                 T1[i].contact_no,
62                 T1[i].enhour, // entry hour
63                 T1[i].enmin); // entry minutes
64
65     }
66
67     fclose(fp);
68     printf("T1 saved successfully to dat1.txt\n");
69 }
70
71 void save_T2(s2 T2[], int count) {
72     FILE *fp = fopen("dat2.txt", "w");
73     if (fp == NULL) {
74         printf("Error opening file \n");
75         return;
76     }
77
78
79     for (int i = 0; i < count; i++)
80     {
81         fprintf(fp, "%d %s %s %d %s %s %s %d %d\n",
82                 T2[i].sno,
83                 T2[i].date,
84                 T2[i].tower_no,
85                 T2[i].slot_no,
86                 T2[i].status,
87                 T2[i].carno,
88                 T2[i].contact_no,
89                 T2[i].enhour,
90                 T2[i].enmin);
91     }
92
93     fclose(fp);
94
95     printf("T2 saved successfully to dat2.txt \n");
96 }
97 // Function for EXIT TIME CALCULATION & SLIP GENERATION
98
99
100 void calc_rev(int checkin_h, int checkin_m, int checkout_h, int checkout_m) {
101
102     int entry_minutes = checkin_h * 60 + checkin_m;
103     int exit_minutes = checkout_h * 60 + checkout_m;
104
105     int duration = exit_minutes - entry_minutes;

```

```

105     int duration = exit_minutes - entry_minutes;
106
107     if (duration <= 0) {
108         printf("Invalid time input\n");
109         return;
110     }
111
112     int payment = 0;
113     if (duration <= 60)
114         payment = 50;
115     else if (duration <= 180)
116         payment = 80;
117     else
118         payment = 80 + (duration - 180) * 2;
119
120     //Final SLIP GENERATED
121
122     printf("\n----- PARKING PAYMENT -----");
123     printf("Entry Time : %d %d\n", checkin_h, checkin_m);
124     printf("Exit Time : %d %d\n", checkout_h, checkout_m);
125     printf("Duration : %d minutes\n", duration);
126     printf("Payment : %d\n", payment);
127     printf("-----\n");
128 }

131 int main() {
132     s1 T1[100];
133     s2 T2[100];
134     FILE *fptr;
135     int a = 0;
136     int b = 0;
137
138
139     // MAIN MENU STARTED HERE
140     for (int i = 0; i < 100; i++) {
141         T1[i].sno = i + 1;
142         strcpy(T1[i].date, "NA");
143         strcpy(T1[i].tower_no, "T1");
144         T1[i].slot_no = i + 1;
145         strcpy(T1[i].status, "EMPTY");
146         strcpy(T1[i].carno, "NA");
147         strcpy(T1[i].contact_no, "NA");
148         T1[i].enhour = 0;
149         T1[i].enmin = 0;
150
151         T2[i].sno = i + 1;
152         strcpy(T2[i].date, "NA");
153         strcpy(T2[i].tower_no, "T2");
154         T2[i].slot_no = i + 1;
155         strcpy(T2[i].status, "EMPTY");
156         strcpy(T2[i].carno, "NA");

```

```

157         strcpy(T2[i].contact_no, "NA");
158         T2[i].enhour = 0;
159         T2[i].enmin = 0;
160     }
161
162     fptr = fopen("dat1.txt", "r");
163     if (fptr != NULL)
164     {
165         while (fscanf(fptr, "%d %19s %4s %d %19s %19s %19s %d %d",
166                         &T1[a].sno,
167                         T1[a].date,
168                         T1[a].tower_no,
169                         &T1[a].slot_no,
170                         T1[a].status,
171                         T1[a].carno,
172                         T1[a].contact_no,
173                         &T1[a].enhour,
174                         &T1[a].enmin) == 9)
175         {
176             a++;
177             if (a >= 100) break;
178         }
179         fclose(fptr);
180     } else {
181
182     }
183
184     fptr = fopen("dat2.txt", "r");
185
186     if (fptr != NULL) {
187         while (fscanf(fptr, "%d %19s %4s %d %19s %19s %19s %d %d",
188                         &T2[b].sno,
189                         T2[b].date,
190                         T2[b].tower_no,
191                         &T2[b].slot_no,
192                         T2[b].status,
193                         T2[b].carno,
194                         T2[b].contact_no,
195                         &T2[b].enhour,
196                         &T2[b].enmin) == 9)
197         {
198             b++;
199             if (b >= 100) break;
200         }
201         fclose(fptr);
202     } else {
203
204     }
205
206     srand((unsigned int)time(NULL));
207
208     //Main Menue
209     int choice;
210     int i = 1;

```

```

210     int j = 1;
211     while (j == 1) {
212         printf("\n");
213         printf("\n");
214
215         printf(" \t ----- PARKING MANAGEMENT SYSTEM ----- \n");
216         printf("\n");
217         printf("1. Admin Mode\n");
218         printf("2. Manual Parking System\n");
219         printf("3. Automatic Parking System\n");
220         printf("4. Revenue Collection Report\n");
221         printf("5. Exit \n");
222         printf("\n");
223         printf("Enter your choice: ");
224         if (scanf("%d", &choice) != 1) {
225             while (getchar() != '\n');
226             continue;
227         }
228
229         switch (choice) {
230             // ADMIN MODE
231
232         case 1:
233             {
234                 char pass[50];
235                 printf("\n");
236                 printf("ENTER PASSWORD : ");
237                 scanf("%s", pass);
238                 // password = admin
239
240                 if (strcmp(pass, "admin") == 0)
241                 {
242                     struct1(T1);
243                     struct2(T2);
244                 }
245                 else
246                 {
247                     printf("\n");
248                     printf("wrong password\n");
249                     printf("Try Again \n");
250                     printf("\n");
251                 }
252
253                 break;
254             }
255             // Manual Parking System
256
257         case 2:
258             {
259                 int t, slot, aslot;
260                 while (1) {
261                     printf("Enter 2 wheeler or 4 wheeler (enter 2 or 4): ");
262                     if (scanf("%d", &t) != 1) { while (getchar() != '\n'); break; }
263
264                     if (t == 2) {
265                         for (int i = 0; i < 99; i++) {
266                             if (strcmp(T1[i].status, "EMPTY") == 0) {
267                                 printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO.: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Entry Mi
268                                     T1[i].sno, T1[i].date, T1[i].tower_no, T1[i].slot_no, T1[i].status, T1[i].carno, T1[i].contact_no, T1[i].enhour, T1[i].enmin);
269                             }
270                         }
271                     printf("Enter Slot no: ");
272                     scanf("%d", &slot);
273
274                     for (int j2 = 0; j2 < 99; j2++) {
275                         if (T1[j2].slot_no == slot) {
276                             aslot = slot - 1;
277                             break;
278                         }
279                     }
280
281                     //Customer details input for T1
282
283                     printf("Enter the following details:\n");

```

```

283         //Customer details input for T1
284
285         printf("Enter the following details:\n");
286
287         printf("ENTER THE DATE(DD-MM-YYYY) : ");
288         scanf("%s", T1[aslot].date);
289
290         printf("Enter contact no: ");
291         scanf("%s", T1[aslot].contact_no);
292
293         strcpy(T1[aslot].status, "FULL");
294
295         printf("Enter car no: ");
296         scanf("%s", T1[aslot].carno);
297
298         printf("Enter entry hour: \n");
299         scanf("%d", &T1[aslot].enhour);
300
301         printf("Enter entry min: \n");
302         scanf("%d", &T1[aslot].enmin);
303
304         save_T1(T1, 99);
305
306         printf("Slot %d Updated Successfully \n\n", slot);
307     }
308
309     if (t == 4)
310     {
311         for (int i = 0; i < 99; i++) {
312             if (strcmp(T2[i].status, "EMPTY") == 0) {
313                 printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Entry Min : %d\n",
314                         T2[i].sno, T2[i].date, T2[i].tower_no, T2[i].slot_no, T2[i].status, T2[i].carno, T2[i].contact_no, T2[i].enhour, T2[i].enmin);
315             }
316         }
317         printf("Enter Slot no: ");
318         scanf("%d", &slot);
319
320         for (int j2 = 0; j2 < 99; j2++) {
321             if (T2[j2].slot_no == slot) {
322                 aslot = slot - 1;
323                 break;
324             }
325         }
326     }
327
328     //Customer details input for T2
329
330     printf("Enter the following details:\n");
331
332     printf("Enter date: \n");

```

```

332     printf("Enter date. \n");
333     scanf("%s", T2[aslot].date);
334
335     printf("Enter contact no: \n");
336     scanf("%s", T2[aslot].contact_no);
337
338     strcpy(T2[aslot].status, "FULL");
339
340     printf("Enter car no: ");
341     scanf("%s", T2[aslot].carno);
342
343     printf("Enter entry hour : \n");
344     scanf("%d", &T2[aslot].enhour);
345
346     printf("Enter entry min : \n");
347     scanf("%d", &T2[aslot].enmin);
348
349
350     save_T2(T2, 99);
351
352     printf("Slot %d Updated Successfully \n\n", slot);
353 }
354 {
355     int Ch;

```

```

358     printf("Enter 1 for yes \n");
359     printf("Enter 2 for no \n");
360     scanf("%d", &Ch);
361
362     if (Ch == 2) {
363         break;
364     }
365
366
367     ak;
368
369     MATIC PARKING SYSTEM
370 {
371     le(1){
372         int sensor = rand() % 2;
373         if (sensor == 1){
374             printf("\n");
375             printf("Vehicle detected at the entry gate \n");
376             printf("\n");
377
378             int t,slot,aslot;
379             while (1)
380             {
381                 printf("Enter 2 wheeler or 4 wheeler (2/4): ");
382                 if (scanf("%d", &t) != 1) { while (getchar() != '\n'); break; }

```

```

386     if (t == 2) {
387         for (int i = 0; i < 99; i++) {
388             if (strcmp(T1[i].status, "EMPTY") == 0) {
389                 printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO.: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Ent
390                                         T1[i].sno, T1[i].date, T1[i].tower_no, T1[i].slot_no, T1[i].status, T1[i].carno, T1[i].contact_no, T1[i].enhour, T1[i].enmin);
391             }
392         }
393         printf("Enter Slot no: ");
394         scanf("%d", &slot);
395
396         for (int j2 = 0; j2 < 99; j2++) {
397             if (T1[j2].slot_no == slot) {
398                 aslot = slot - 1;
399                 break;
400             }
401         }
402
403         printf("Enter the following details:\n");
404
405         printf("ENTER THE DATE: ");
406         scanf("%s", T1[aslot].date);
407
408         printf("Enter contact no: ");
409         scanf("%s", T1[aslot].contact_no);
410
411
412         strcpy(T1[aslot].status, "FULL");
413
414         printf("Enter car no: ");
415         scanf("%s", T1[aslot].carno);
416
417
418         printf("Enter entry hour: \n");
419         scanf("%d", &T1[aslot].enhour);
420
421         printf("Enter entry min: \n");
422         scanf("%d", &T1[aslot].enmin);
423
424
425         save_T1(T1, 99);
426         printf("Slot %d Updated Successfully\n\n", slot);
427     }
428
429
430
431     if (t == 4) {
432
433         for (int i = 0; i < 99; i++) {
434             if (strcmp(T2[i].status, "EMPTY") == 0) {
435                 printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO.: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Entry Min : %
436                                         T2[i].sno, T2[i].date, T2[i].tower_no, T2[i].slot_no, T2[i].status, T2[i].carno, T2[i].contact_no, T2[i].enhour, T2[i].enmin);
437             }
438         }
439         printf("Enter Slot no: ");
440         scanf("%d", &slot);
441
442         for (int j2 = 0; j2 < 99; j2++) {
443             if (T2[j2].slot_no == slot) {
444                 aslot = slot - 1;
445                 break;
446             }
447         }
448
449
450         printf("Enter the following details:\n");
451
452         printf("ENTER THE DATE: ");
453         scanf("%s", T2[aslot].date);
454
455

```

```

451     printf("Enter the following details:\n");
452
453     printf("ENTER THE DATE: ");
454     scanf("%s", T2[aslot].date);
455
456     printf("Enter contact no: ");
457     scanf("%s", T2[aslot].contact_no);
458
459     strcpy(T2[aslot].status, "FULL");
460
461     printf("Enter car no: ");
462     scanf("%s", T2[aslot].carno);
463
464
465     printf("Enter entry hour: \n");
466     scanf("%d", &T2[aslot].enhour);
467
468     printf("Enter entry min: \n");
469     scanf("%d", &T2[aslot].enmin);
470
471     save_T2(T2, 99);
472
473     printf("Slot %d Updated Successfully\n\n", slot);
474
475
476     break;
477 }

480     else{
481         printf("\n");
482         printf("SORRY \n");
483         printf("No vehicle detected \n");
484         printf("\n");
485     }
486     int Ch;
487     printf("Want to continue:\n ");
488     printf("Enter 1 for yes \n");
489     printf("Enter 2 for no \n");
490     scanf("%d", &Ch);
491     if (Ch == 2) {
492         break;
493     }
494
495     break;
496
497 // Revenue Collection Report
498 case 4:
499
500     //exit
501
502     int x, eslot = -1;
503     int exhour, exmin;
504     char arr[20];

```

```

504     char arr[30];
505     printf("Enter 2 or 4 wheeler (2/4): ");
506     scanf("%d", &x);
507
508     if(x==2)
509     {
510         printf("Enter vehicle number: ");
511         scanf("%s", arr);
512         for(int i=0;i<99;i++)
513         {
514             if(strcmp(T1[i].carno, arr)==0)
515             {
516                 eslot = i;
517                 break;
518             }
519             if (eslot == -1) {
520                 printf("Vehicle not found in T1\n");
521                 break;
522             }
523             printf("Enter exit hour: \n");
524             scanf("%d", &exhour);
525
526             printf("Enter exit min: \n");
527             scanf("%d", &exmin);

528
529             printf("Enter exit hour: \n");
530             scanf("%d", &exhour);

531             printf("Enter exit min: \n");
532             scanf("%d", &exmin);

533
534             calc_rev(T1[eslot].enhour,T1[eslot].enmin, exhour,exmin);
535
536             strcpy(T1[eslot].contact_no, "NA");
537             strcpy(T1[eslot].carno, "NA");
538             strcpy(T1[eslot].date, "NA");
539             strcpy(T1[eslot].status, "EMPTY");
540             T1[eslot].enhour = 0;
541             T1[eslot].enmin = 0;

542             save_T1(T1, 99);

543             break;
544         }
545     }

```

```

546         printf("Enter vehicle number: ");
547         scanf("%s", arr);
548         for(int i=0;i<99;i++) {
549             if(strcmp(T2[i].carno, arr)==0) {
550                 eslot = i;
551                 break;
552             }
553         }
554         if (eslot == -1) {
555             printf("Vehicle not found in T2\n");
556             break;
557         }
558         printf("Enter exit hour: \n");
559         scanf("%d", &exhour);
560
561         printf("Enter exit min: \n");
562         scanf("%d", &exmin);
563
564
565         calc_rev(T2[eslot].enhour,T2[eslot].enmin , exhour, exmin);
566         strcpy(T2[eslot].contact_no, "NA");
567         strcpy(T2[eslot].carno, "NA");
568         strcpy(T2[eslot].date, "NA");
569         strcpy(T2[eslot].status, "EMPTY");
570
571         T2[eslot].enhour = 0;
572         T2[eslot].enmin = 0;
573
574         save_T2(T2, 99);
575
576         break;
577     }
578
579     break;
580
581 // BREAK
582 case 5:
583 {
584     printf("-----\n");
585     printf("THANKYOU FOR USING OUR PARKING MANAGEMENT SYSTEM\n");
586     printf("-----\n");
587
588     return 0;
589     break;
590
591 default:
592     printf("Invalid choice\n");
593     break;
594 }
595
596 return 0;
597 }
598
599 // display functions for both the structures
600 void struct1(s1 T1[]) {
601     printf("\n\n");
602     printf("-----TABLE FOR T1-----\n");
603     printf("\n");
604     for(int i=0;i<99;i++) {
605         printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO.: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Entry Min : %d\n",
606               T1[i].sno, T1[i].date, T1[i].tower_no, T1[i].slot_no, T1[i].status, T1[i].carno, T1[i].contact_no, T1[i].enhour, T1[i].enmin);
607         printf("\n");
608     }
609     printf("-----\n");
610 }
611
612 void struct2(s2 T2[]) {
613     printf("\n\n");
614     printf("-----TABLE FOR T2-----\n");
615     printf("\n");
616     for(int i=0;i<99;i++) {
617         printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO.: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Entry Min : %d \n",
618               T2[i].sno, T2[i].date, T2[i].tower_no, T2[i].slot_no, T2[i].status, T2[i].carno, T2[i].contact_no, T2[i].enhour, T2[i].enmin);
619         printf("\n");
620     }
621     printf("-----\n");
622 }
623
624 void struct2(s2 T2[]) {
625     printf("\n\n");
626     printf("-----TABLE FOR T2-----\n");
627     printf("\n");
628     for(int i=0;i<99;i++) {
629         printf("Sr No : %d || Date: %s || TOWER_NO: %s || SLOT_NO.: %d || STATUS: %s || CAR_NO: %s || CONTACT_NO: %s || Entry Hour : %d || Entry Min : %d \n",
630               T2[i].sno, T2[i].date, T2[i].tower_no, T2[i].slot_no, T2[i].status, T2[i].carno, T2[i].contact_no, T2[i].enhour, T2[i].enmin);
631         printf("\n");
632     }
633     printf("-----\n");

```

# **Output screenshots with description:**

- Here we are running the code in admin mode .

```
*C:\Users\User\Desktop\PBL! + - X

Sr No : 12 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 12 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 13 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 13 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 14 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 14 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 15 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 15 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 16 || Date: 12-11-2023 || TOWER_NO: T1 || SLOT_NO.: 16 || STATUS: FULL || CAR_NO: GSDYJR12 || CONTACT_NO: 7429837508 || Entry Hour : 12 || Entry Min : 23
Sr No : 17 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 17 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 18 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 18 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 19 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 19 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 20 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 20 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 21 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 21 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 22 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 22 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 23 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 23 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 24 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 24 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 25 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 25 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 26 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 26 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 27 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 27 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 28 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 28 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 29 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 29 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 30 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 30 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 31 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 31 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
```

```
C:\Users\User\Desktop\PBL5 + 
Sr No : 85 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 85 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 86 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 86 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 87 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 87 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 88 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 88 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 89 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 89 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 90 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 90 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 91 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 91 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 92 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 92 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 93 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 93 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 94 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 94 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 95 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 95 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 96 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 96 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 97 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 97 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 98 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 98 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 99 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 99 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 100 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 100 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
-----
----- TABLE FOR T2 -----
Sr No : 1 || Date: NA || TOWER_NO: T2 || SLOT_NO.: 1 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 2 || Date: NA || TOWER_NO: T2 || SLOT_NO.: 2 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
```



- Manual mode chosen and we can see all empty slots now we will update the slots .



```
Sr No : 96 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 96 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 97 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 97 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 98 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 98 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Sr No : 99 || Date: NA || TOWER_NO: T1 || SLOT_NO.: 99 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0
Enter Slot no: 1
Enter the following details:
ENTER THE DATE(DD-MM-YYYY): 12-11-25
Enter contact no: 9834789038
Enter car no: DL45AB1007
Enter entry hour:
12
Enter entry min:
07
T1 saved successfully to dat1.txt
Slot 1 Updated Successfully

Want to continue:
Enter 1 for yes
Enter 2 for no
2
```

- Here we can see the updated records

```
-- PARKING MANAGEMENT SYSTEM --
1. Admin Mode
2. Manual Parking System
3. Automatic Parking System
4. Revenue Collection Report
5. Exit

Enter your choice: 1
ENTER PASSWORD : admin

--TABLE FOR T1--
Sr No : 1 || Date: 12-11-25 || TOWER_NO: T1 || SLOT_NO.: 1 || STATUS: FULL || CAR_NO: DL45AB1007 || CONTACT_NO: 9834789038 || Entry Hour : 12 || Entry Min :
7
```

- Automatic mode chosen here

PARKING MANAGEMENT SYSTEM

1. Admin Mode
  2. Manual Parking System
  3. Automatic Parking System
  4. Revenue Collection Report
  5. Exit

Enter your choice: 3

Vehicle detected at the entry gate

Sr No	Date	TOWER_NO	TOWER_NO	SLOT_NO	SLOT_NO	STATUS	CAR_NO	CONTACT_NO	Entry Hour	Entry Min
15	NA	TOWER_NO	T2	15	15	EMPTY	NA	NA	0	0
16	NA	TOWER_NO	T2	16	16	EMPTY	NA	NA	0	0
17	NA	TOWER_NO	T2	17	17	EMPTY	NA	NA	0	0
18	NA	TOWER_NO	T2	18	18	EMPTY	NA	NA	0	0
19	NA	TOWER_NO	T2	19	19	EMPTY	NA	NA	0	0
20	NA	TOWER_NO	T2	20	20	EMPTY	NA	NA	0	0
21	NA	TOWER_NO	T2	21	21	EMPTY	NA	NA	0	0
22	NA	TOWER_NO	T2	22	22	EMPTY	NA	NA	0	0
23	NA	TOWER_NO	T2	23	23	EMPTY	NA	NA	0	0
24	NA	TOWER_NO	T2	24	24	EMPTY	NA	NA	0	0
25	NA	TOWER_NO	T2	25	25	EMPTY	NA	NA	0	0
26	NA	TOWER_NO	T2	26	26	EMPTY	NA	NA	0	0
27	NA	TOWER_NO	T2	27	27	EMPTY	NA	NA	0	0
28	NA	TOWER_NO	T2	28	28	EMPTY	NA	NA	0	0
29	NA	TOWER_NO	T2	29	29	EMPTY	NA	NA	0	0
30	NA	TOWER_NO	T2	30	30	EMPTY	NA	NA	0	0
31	NA	TOWER_NO	T2	31	31	EMPTY	NA	NA	0	0
32	NA	TOWER_NO	T2	32	32	EMPTY	NA	NA	0	0
33	NA	TOWER_NO	T2	33	33	EMPTY	NA	NA	0	0
34	NA	TOWER_NO	T2	34	34	EMPTY	NA	NA	0	0
35	NA	TOWER_NO	T2	35	35	EMPTY	NA	NA	0	0
36	NA	TOWER_NO	T2	36	36	EMPTY	NA	NA	0	0
37	NA	TOWER_NO	T2	37	37	EMPTY	NA	NA	0	0
38	NA	TOWER_NO	T2	38	38	EMPTY	NA	NA	0	0
39	NA	TOWER_NO	T2	39	39	EMPTY	NA	NA	0	0
40	NA	TOWER_NO	T2	40	40	EMPTY	NA	NA	0	0
41	NA	TOWER_NO	T2	41	41	EMPTY	NA	NA	0	0
42	NA	TOWER_NO	T2	42	42	EMPTY	NA	NA	0	0
43	NA	TOWER_NO	T2	43	43	EMPTY	NA	NA	0	0
44	NA	TOWER_NO	T2	44	44	EMPTY	NA	NA	0	0
45	NA	TOWER_NO	T2	45	45	EMPTY	NA	NA	0	0
46	NA	TOWER_NO	T2	46	46	EMPTY	NA	NA	0	0
47	NA	TOWER_NO	T2	47	47	EMPTY	NA	NA	0	0
48	NA	TOWER_NO	T2	48	48	EMPTY	NA	NA	0	0
49	NA	TOWER_NO	T2	49	49	EMPTY	NA	NA	0	0
50	NA	TOWER_NO	T2	50	50	EMPTY	NA	NA	0	0
51	NA	TOWER_NO	T2	51	51	EMPTY	NA	NA	0	0
52	NA	TOWER_NO	T2	52	52	EMPTY	NA	NA	0	0
53	NA	TOWER_NO	T2	53	53	EMPTY	NA	NA	0	0
54	NA	TOWER_NO	T2	54	54	EMPTY	NA	NA	0	0
55	NA	TOWER_NO	T2	55	55	EMPTY	NA	NA	0	0

```
SLT_NO : 99 || Date: NA || TOWER_NO: 12 || SLOT_NO.: 99 || STATUS: EMPTY || CAR_NO: NA || CONTACT_NO: NA || Entry Hour : 0 || Entry Min : 0  
Enter Slot no: 1  
Enter the following details:  
ENTER THE DATE: 13-11-25  
Enter contact no: 9873609888  
Enter car no: DL686AD79  
Enter entry hour:  
19  
Enter entry min:  
30  
T2 saved successfully to dat2.txt  
Slot 1 Updated Successfully  
  
Want to continue:  
Enter 1 for yes  
Enter 2 for no  
2
```

- Slot updated

-----TABLE FOR T2-----  
Sr No : 1 || Date: 13-11-25 || TOWER\_NO: T2 || SLOT\_NO.: 1 || STATUS: FULL || CAR\_NO: DL686AD79 || CONTACT\_NO: 9873609888 || Entry Hour : 19 || Entry Min : 30

- Revenue is calculated

```
----- PARKING MANAGEMENT SYSTEM -----  
1. Admin Mode  
2. Manual Parking System  
3. Automatic Parking System  
4. Revenue Collection Report  
5. Exit  
  
Enter your choice: 4  
Enter 2 or 4 wheeler (2/4): 2  
Enter vehicle number: DL45AB1007  
Enter exit hour:  
13  
Enter exit min:  
23  
  
----- PARKING BILL -----  
Entry Time : 12 07  
Exit Time : 13 23  
Duration : 76 minutes  
Payment : 80  
  
T1 saved successfully to dat1.txt
```

- We have finally exited the program

```
----- PARKING MANAGEMENT SYSTEM -----  
1. Admin Mode  
2. Manual Parking System  
3. Automatic Parking System  
4. Revenue Collection Report  
5. Exit  
  
Enter your choice: 5  
  
THANKYOU FOR USING OUR PARKING MANAGEMENT SYSTEM  
  
-----  
Process returned 0 (0x0) execution time : 372.398 s  
Press any key to continue.
```

## **CONCLUSION**

### **Summary & achievement of Objectives:**

The Parking Database Management System successfully streamlines the entire parking process from entry to exit. By using basic C programming concepts, the project achieves its main objectives—reducing waiting time, improving accuracy in slot allocation, and making the overall parking experience more organized. The system allows both manual and automatic access, calculates parking charges based on duration, and stores all records securely through file handling. With its simple interface and automated operations, the system proves effective in minimizing human errors and ensuring smooth parking management in busy locations such as malls, hospitals, and corporate offices.

Overall, the project demonstrates how technology can be used in a practical way to help solve real-life problems while fulfilling all the goals set at the beginning of the development.

### **Future Work and Recommendations:**

Although the current version of the Parking Database Management System meets its basic purpose, there are many possibilities for future improvement. The system could be enhanced with features like real-time sensor-based detection of occupied and vacant slots, online booking of parking spaces, and integration with mobile apps for even faster and more convenient access. Adding digital payment

gateways and security features such as vehicle number plate recognition could further improve efficiency and safety.

With more advanced programming techniques and the use of modern technologies, this system can evolve into a complete smart parking solution suitable for large-scale parking facilities and smart cities.

## **REFERENCES**

### **List of all sources cited in the report:**

- 1) Research articles and online resources on parking management and automation for conceptual understanding.
- 2) [www.geeksforgeeks.org](http://www.geeksforgeeks.org) -C Programming Concepts and Examples
- 3)[www.studytonight.com](http://www.studytonight.com) -File Handling in C and Other Core Concepts