Parking Lot Ticketing System Major Project – C Programming

Submitted by: Pranav Singh

Sap Id: 590022659. Batch: B48

Course: B.Tech in Computer Science and Engineering. Semester 1

Course Code: CSEG1041

Submitted to: Mr. Mohsin F. Dar

Academic Year: 2025

Abstract: -

This project is a simple parking lot management system developed using the C programming language. It allows vehicles to be entered and exited with timestamps, calculates the parking fee based on parking duration, and displays currently parked vehicles.

The project uses file handling to store data permanently and structures to hold vehicle details. This system helps automate basic parking operations without manual records, making it easy to manage vehicle flow and daily revenue.

Problem Definition: -

Manual vehicle entry in parking areas often causes confusion, missing records, and errors in fee calculation.

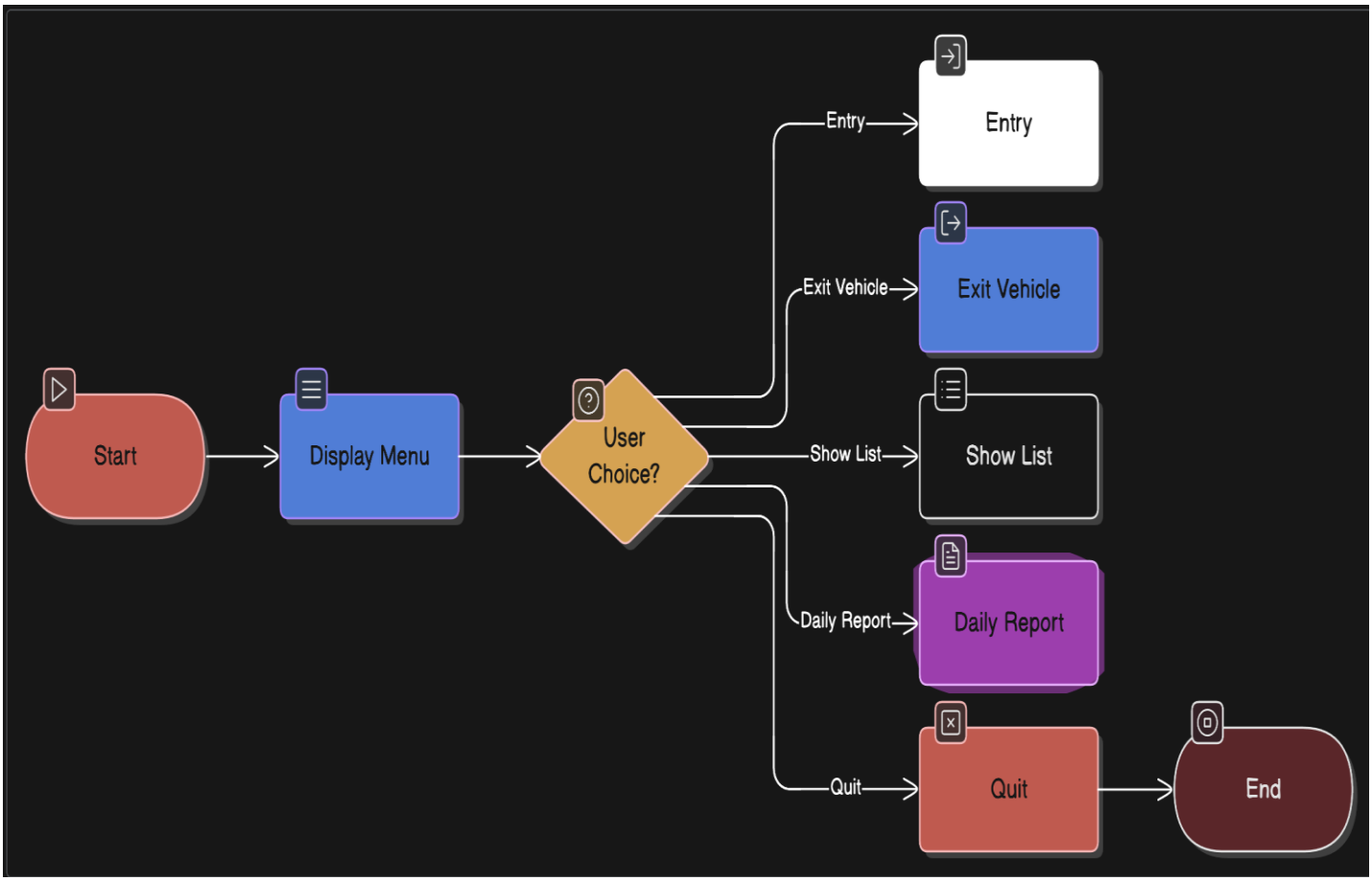
Keeping track of in/out time and calculating charges manually is time-consuming.

Therefore, we designed a digital parking system that stores vehicle details, calculates fees automatically, and generates daily reports.

This program improves parking management and reduces human error.

System Design: -

Flowchart-



Algorithms: -

Algorithm for Vehicle Entry

1. Generate next ticket ID
2. Ask user for vehicle number & type
3. Store entry time (time (NULL))
4. Write record to file
5. Print ticket issued message

Algorithm for Exit

1. Ask for vehicle number
2. Search file to match entry
3. Store exit time
4. Calculate fee using time difference
5. Update file and print receipt

Fee Calculation Logic

- First 30 minutes free
- After that charged hourly
- 2W = ₹10/hour
- 4W = ₹20/hour

Minimum charge = ₹10

Implementation: -

The program is divided into two files — main.c and parking.h.

The header file contains structure definition and function prototypes. The main file contains complete logic for menu, entry, exit, and reporting.

File handling is used (fopen, fwrite, fread), so all parking data remains stored even after closing the program.

Functions are used to separate tasks such as in (), out Vehicle (), show (), and report () for clean modular implementation.

Snippets

1 – Structure to store vehicle details

```c
typedef struct {
    int id;
    char num[20];
    int type;       // 1=2W, 2=4W
    time_t in, out;
    float fee;
    int status;     // 1=in, 2=out
} Data;
```

2 – Function to generate next ID

```c
int getID() {
    FILE *fp = fopen(DATA_FILE, "rb");
    if (!fp) return 1;

    Data d;
    int mx = 0;
    while (fread(&d, sizeof(d), 1, fp) == 1) {
        if (d.id > mx) mx = d.id;
    }
    fclose(fp);
    return mx + 1;
}
```

3 - Parking fee calculation logic

```c
float calc(time_t a, time_t b, int t) {
    double mins = difftime(b, a) / 60.0;
    if (mins <= 30) return 0;       // first 30 min free

    double h = (mins - 30) / 60.0;
    int x = (int)(h + 1);           // round up to next hour

    int rate = (t == 1 ? 10 : 20); // 2W = 10, 4W = 20
    float total = x * rate;
    if (total < 10) total = 10;     // minimum 10
    return total;
}
```

4 - Vehicle Entry Details

```c
void in() {
    FILE *fp = fopen(DATA_FILE, "ab");
    if (!fp) {
        printf("file err\n");
        return;
    }

    Data d;
    d.id =
    printf(
    scanf("
    printf("Type 1=2W 2=4W: ");
    scanf("%d", &d.type);

    d.in = time(NULL);
    d.out = 0;
    d.fee = 0;
    d.status = 1;       // parked

    fwrite(&d, sizeof(d), 1, fp);
    fclose(fp);

    printf("\nTicket %d added!\n", d.id);
}
```

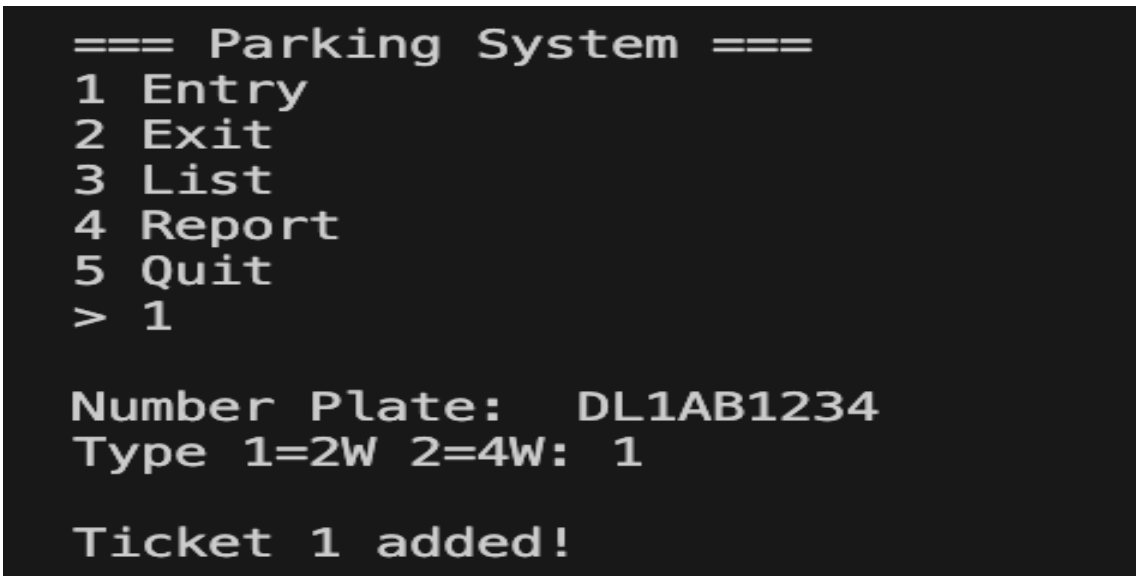
(char [17])"Type 1=2W 2=4W: "

Generate Copilot summary

Testing And Result: -

Test Case	Input	Expected Output	Result
Add 2-wheeler	Entry → 1, type 1	Ticket issued	Pass
Exit same vehicle	Exit → plate	Fee calculated	Pass
Invalid exit	Exit → unknown plate	Show not found	Pass
Daily report	Date given	Show total revenue	Pass

Terminal Outputs: -



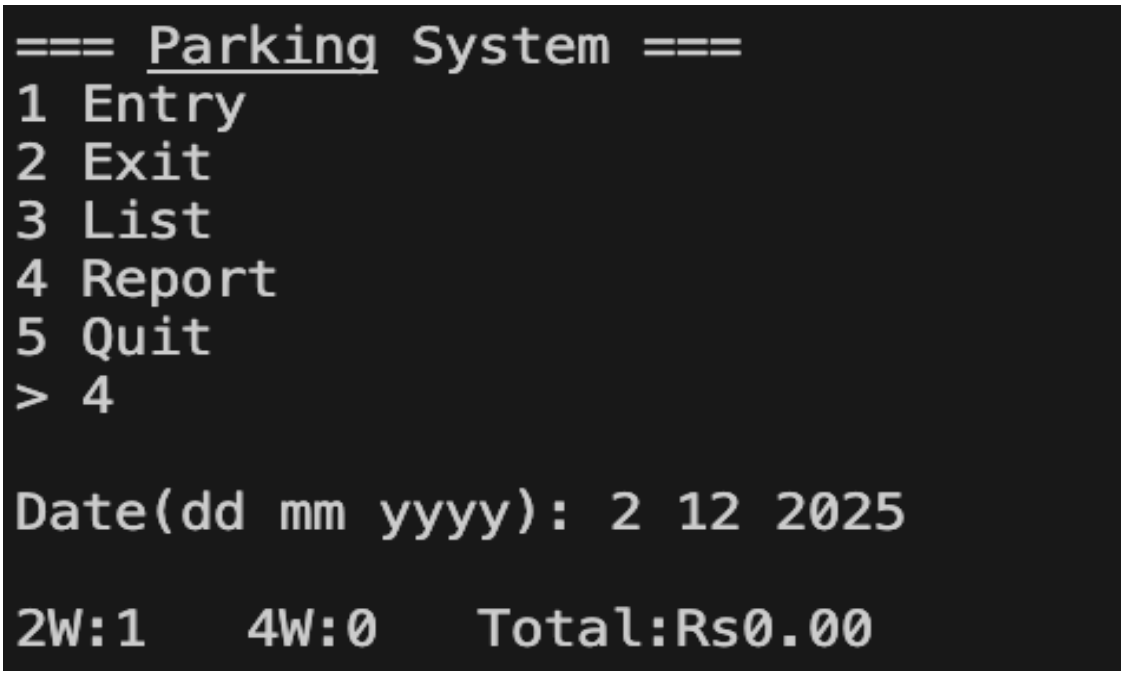
```
=== Parking System ===
1 Entry
2 Exit
3 List
4 Report
5 Quit
> 2

Enter Plate:  DL1AB1234

Exit Done.
Ticket:1 Fee:Rs0.00 Time:24.1 mins
```

```
=== Parking System ===
1 Entry
2 Exit
3 List
4 Report
5 Quit
> 2

Enter Plate: AAF4567
Not inside.
```



Conclusion: -

This project successfully automates parking management using C programming.

It reduces manual work, stores data permanently, and simplifies fee calculation.

Further improvements like online payment, slot availability, and camera scanning could be added in future versions.

References: -

- Class notes
- C programming book (E. Balagurusamy / Let Us C)
- Online documentation for stdio.h & time.h

Appendix: -

Source Code Repository:

https://github.com/PranavSingh1729/parking-lot-c-project.git