PROJECT REPORT

---

1. Title Page

PROJECT REPORT
Petrol Pump Management System
Course Code: CSEG1032
Course Title: Programming in C
Student Name: Yuvraj Dhiman
SAP ID: 590025374
Semester: 1st Year

---

2. Abstract

The Petrol Pump Management System is a C-based console project designed to automate and simplify fuel management operations. The system records fuel types, quantity, billing, and transaction details. It reduces manual effort, minimizes errors, and improves efficiency. This project demonstrates the use of fundamental C programming concepts such as functions, loops, structures, and file handling.

---

3. Problem Definition

Managing daily operations at a petrol pump manually can lead to errors, inconsistencies, and time consumption. This project aims to provide a computerized system that:

- Manages fuel records efficiently.
- Calculates bills automatically based on fuel type and quantity.
- Stores transaction details.
- Ensures accuracy and faster processing.

---

4. System Design

Flowchart

1. Start → Display Menu → Select Fuel Type → Enter Quantity → Calculate Bill → Show Summary → Store Record → End.

Algorithm

1. Start program.
2. Display list of available fuels with price.
3. Ask user to select fuel type.
4. Input required fuel quantity.
5. Calculate total cost using formula: **amount = price × quantity**.
6. Display final bill.
7. Store transaction in a file.
8. End program.

---

5. Implementation Details (with simple code snippet)

5.1 Header File — sales.h
Purpose:
This file stores structure definitions and function declarations used across the program.

```
typedef struct {
    int id;
    char date[11];
    char vehReg[20];
    char driver[30];
    char fuel[10];
    double liters;
    double rate;
    double total;
} Sale;
```

-This structure stores **all details of a fuel sale** in one place.

5.2 Getting Today's Date
Purpose:
Automatically generates today's date so the program knows which sales happened today.

```
char* today_date() {
    static char d[11];
    time_t t = time(NULL);
    struct tm *now = localtime(&t);

    sprintf(d, "%04d-%02d-%02d",
            now->tm_year + 1900,
            now->tm_mon + 1,
            now->tm_mday);

    return d;
}
```

-This ensures the system works daily without manually entering dates.

5.3 Auto-Incrementing Sale ID

Purpose:

Read last ID from sales.csv and generate the next ID.

```
int next_sale_id() {
    FILE *fp = fopen(SALES_F, "r");
    if (!fp) return 1;

    int id = 1, last = 0;
    char line[200];

    while (fgets(line, sizeof(line), fp)) {
        sscanf(line, "%d,", &last);
        id = last + 1;
    }

    fclose(fp);
    return id;
}
```

-Every new sale gets a unique ID (1, 2, 3, …)

5.4 Saving a Sale

Purpose:

Store sale data permanently into sales.csv.

```
void save_sale(Sale *s) {
    FILE *fp = fopen(SALES_F, "a");

    fprintf(fp, "%d,%s,%s,%s,%s,%.2f,%.2f,%.2f\n",
            s->id, s->date, s->vehReg, s->driver,
            s->fuel, s->liters, s->rate, s->total);

    fclose(fp);
}
```

-Data is saved in comma-separated format (CSV).

5.5 Making a Sale

Purpose:

Takes user input, calculates the bill, generates sale entry.

```
void make_sale() {
    Sale s;

    printf("Vehicle number: ");
    scanf("%s", s.vehReg);

    printf("Driver name: ");
    scanf("%s", s.driver);

    printf("Fuel type (Petrol/Diesel): ");
    scanf("%s", s.fuel);

    printf("Liters: ");
    scanf("%lf", &s.liters);

    if (strcmp(s.fuel, "Petrol") == 0)
        s.rate = 110;
    else if (strcmp(s.fuel, "Diesel") == 0)
        s.rate = 95;
    else
        s.rate = 100;
```

```
    s.total = s.liters * s.rate;

    s.id = next_sale_id();
    strcpy(s.date, today_date());

    save_sale(&s);

    printf("Sale saved! Total = %.2f\n", s.total);
}
```

-This is the main function used by the operator at the petrol pump.
-Automatically calculates total amount.

5.6 Viewing Today's Sales

Purpose:

Displays only the sales that happened today.

```
void view_sales_today() {
    FILE *fp = fopen(SALES_F, "r");
    if (!fp) return;

    char today[11];
    strcpy(today, today_date());

    char line[200];
    int id;
    char date[11], veh[20], driver[30], fuel[10];
    double liters, rate, total;

    printf("Today's Sales (%s):\n", today);

    while (fgets(line, sizeof(line), fp)) {
        sscanf(line, "%d,%10[^,],%19[^,],%29[^,],%9[^,],%lf,%lf,%lf",
               &id, date, veh, driver, fuel, &liters, &rate, &total);

        if (strcmp(date, today) == 0)
            printf("%d  %s  %s  %.2fL  Rs %.2f\n",
                   id, veh, fuel, liters, total);
    }

    fclose(fp);
}
```

-Helps the manager quickly check daily activity.

5.7 Showing Today's Fuel Totals

Purpose:

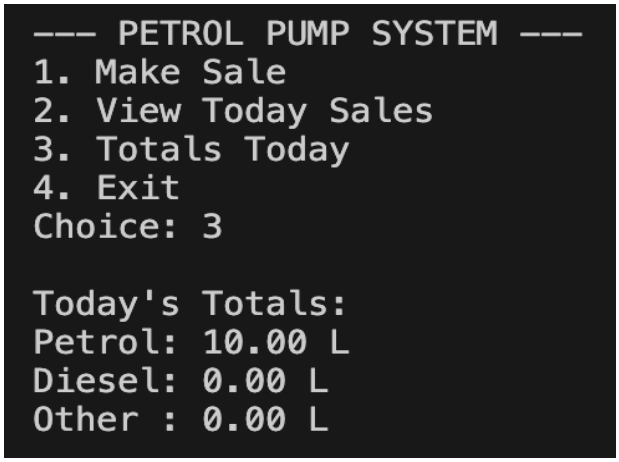
Shows how many liters of Petrol, Diesel, and Other fuels were sold today.

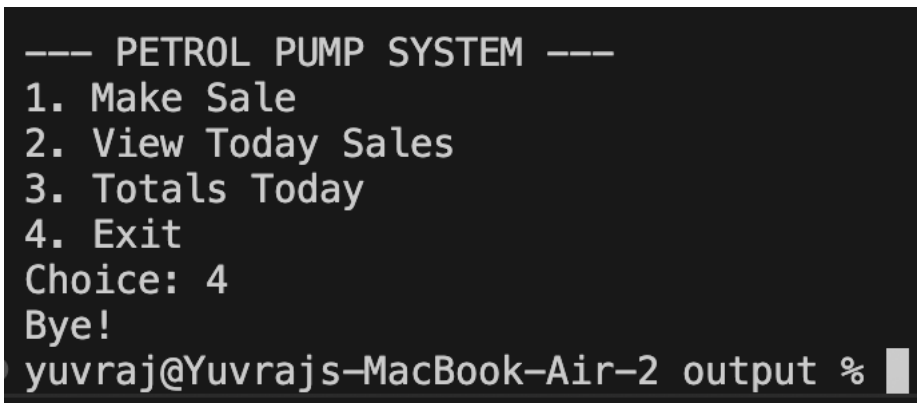
```
void totals_today() {
    FILE *fp = fopen(SALES_F, "r");
    if (!fp) return;

    char today[11];
    strcpy(today, today_date());

    char line[200];
    char date[11], fuel[10];
    double liters;
    int id;

    double petrol = 0, diesel = 0, other = 0;

    while (fgets(line, sizeof(line), fp)) {
        sscanf(line, "%d,%10[^,],%*[^,],%*[^,],%9[^,],%lf",
```

```
                &id, date, fuel, &liters);

        if (strcmp(date, today) == 0) {
            if (strcmp(fuel, "Petrol") == 0)
                petrol += liters;
            else if (strcmp(fuel, "Diesel") == 0)
                diesel += liters;
            else
                other += liters;
        }
    }

    fclose(fp);

    printf("Today's Totals:\n");
    printf("Petrol: %.2f L\n", petrol);
    printf("Diesel: %.2f L\n", diesel);
    printf("Other : %.2f L\n", other);
}
-Useful for inventory management.
```

---

6. Testing & Results. Testing & Results**





7. Conclusion & Future Work

The Petrol Pump Management System simplifies fuel billing and transaction management. It demonstrates the use of core C programming concepts effectively. In the future, the system can be expanded with:

- Graphical User Interface (GUI).
- Real-time inventory tracking.
- Admin login system.
- GST and discount modules.

---

8. References

- Class Notes by **Dr. Tanu Singh**
- **Let Us C** by Yashavant Kanetkar
- **GeeksforGeeks** (www.geeksforgeeks.org)
- Various online C programming documentation