TDPL-ASSIGNMENT→02

k.VAMSHI

422175

CSE-A

19. Catering Management System

Features: Menu planning, event scheduling, order management, and billing.

Extensions: Include customer feedback, dietary restrictions management, and integration with event management platforms.

20. Donation Management System

Features: Donor registration, donation tracking, fund allocation, and receipt generation.

Extensions: Implement donation campaigns, automated thank-you emails, and impact reporting.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_NAME_LENGTH 100
#define MAX_MENU_ITEMS 50
#define MAX_EVENTS 20
#define MAX_ORDERS 100
#define MAX_FEEDBACK 100
#define MAX_DIETARY_RESTRICTIONS 100

typedef struct {
    char itemName[MAX_NAME_LENGTH];
    float price;
} MenuItem;
```

```
typedef struct {
    char eventName[MAX_NAME_LENGTH];
    char eventDate[11]; // YYYY-MM-DD
    MenuItem menu[MAX_MENU_ITEMS];
    int menuSize;
} Event;

typedef struct {
    char customerName[MAX_NAME_LENGTH];
    char eventName[MAX_NAME_LENGTH];
    float totalAmount;
} Order;

typedef struct {
    Order orders[MAX_ORDERS];
    int orderCount;
} Billing;

typedef struct {
    char eventName[MAX_NAME_LENGTH];
    char feedback[MAX_FEEDBACK];
    int rating; // Rating out of 5
} Feedback;

typedef struct {
    char customerName[MAX_NAME_LENGTH];
    char restrictions[MAX_DIETARY_RESTRICTIONS];
} DietaryRestriction;

MenuItem menu[MAX_MENU_ITEMS];
```

```
int menuSize = 0;
void addMenuItem(MenuItem *menu, int *size, const char *itemName, float price);
void createEvent(Event *event, const char *eventName, const char *eventDate);
void addMenuItemToEvent(Event *event, const char *itemName);
void addOrder(Billing *billing, const char *customerName, const char *eventName, float amount);
void generateInvoice(const Billing *billing);
void collectFeedback(Feedback *feedbacks, int *feedbackCount, const char *eventName, const char
*feedback, int rating);
void displayFeedback(const Feedback *feedbacks, int feedbackCount);
void addDietaryRestriction(DietaryRestriction *restrictions, int *restrictionCount, const char
*customerName, const char *restriction);
void displayEvents(const Event *events, int eventCount);
void saveEvents(const Event *events, int eventCount);
void loadEvents(Event *events, int *eventCount);
void displayMenu();


int main() {
   Event events[MAX_EVENTS];
   int eventCount = 0;
   Billing billing = {0};
   Feedback feedbacks[MAX_EVENTS];
   int feedbackCount = 0;
   DietaryRestriction restrictions[MAX_DIETARY_RESTRICTIONS];
   int restrictionCount = 0;



   loadEvents(events, &eventCount);


   int choice;
   char itemName[MAX_NAME_LENGTH];
   char eventName[MAX_NAME_LENGTH];
   char eventDate[11];
   char customerName[MAX_NAME_LENGTH];
   float price;
   float amount;
   char feedback[MAX_FEEDBACK];
   char restriction[MAX_DIETARY_RESTRICTIONS];
   int rating;

   while (1) {
      displayMenu();
      scanf("%d", &choice);
      getchar();

      if (choice == 1) {
         printf("Enter item name: ");
         fgets(itemName, MAX_NAME_LENGTH, stdin);
         itemName[strcspn(itemName, "\n")] = '\0';
         printf("Enter price: ");
         scanf("%f", &price);
         getchar();
         addMenuItem(menu, &menuSize, itemName, price);
         printf("Menu item added.\n");
      } else if (choice == 2) {
         displayFeedback(feedbacks, feedbackCount);
      } else if (choice == 3) {
         printf("Enter event name: ");
         fgets(eventName, MAX_NAME_LENGTH, stdin);
         eventName[strcspn(eventName, "\n")] = '\0';
         printf("Enter event date (YYYY-MM-DD): ");
         fgets(eventDate, 11, stdin);
         eventDate[strcspn(eventDate, "\n")] = '\0';
         createEvent(&events[eventCount], eventName, eventDate);
```

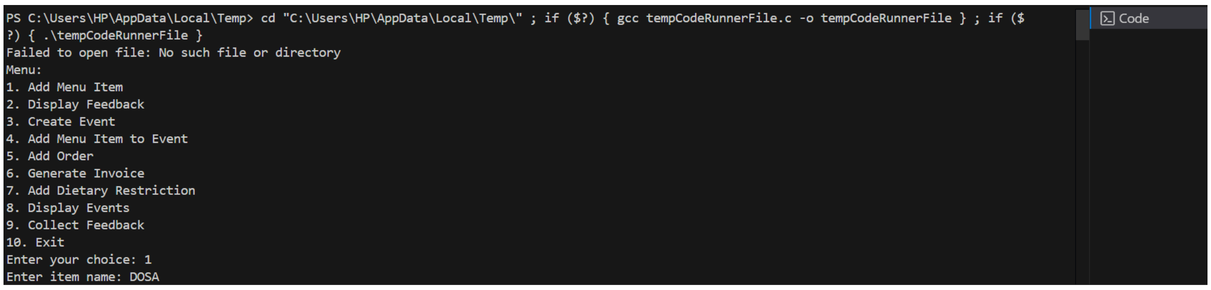
```
        eventCount++;
        printf("Event created.\n");
    } else if (choice == 4) {
        printf("Enter event name: ");
        fgets(eventName, MAX_NAME_LENGTH, stdin);
        eventName[strcspn(eventName, "\n")] = '\0';
        printf("Enter menu item to add: ");
        fgets(itemName, MAX_NAME_LENGTH, stdin);
        itemName[strcspn(itemName, "\n")] = '\0';

        for (int i = 0; i < eventCount; i++) {
            if (strcmp(events[i].eventName, eventName) == 0) {
                addMenuItemToEvent(&events[i], itemName);
                break;
            }
        }
        printf("Menu item added to event.\n");
    } else if (choice == 5) {
        printf("Enter customer name: ");
        fgets(customerName, MAX_NAME_LENGTH, stdin);
        customerName[strcspn(customerName, "\n")] = '\0';
        printf("Enter event name: ");
        fgets(eventName, MAX_NAME_LENGTH, stdin);
        eventName[strcspn(eventName, "\n")] = '\0';
        printf("Enter total amount: ");
        scanf("%f", &amount);
        getchar();
        addOrder(&billing, customerName, eventName, amount);
        printf("Order added.\n");
    } else if (choice == 6) {
        generateInvoice(&billing);
    } else if (choice == 7) {
        printf("Enter customer name: ");
        fgets(customerName, MAX_NAME_LENGTH, stdin);
        customerName[strcspn(customerName, "\n")] = '\0';
        printf("Enter dietary restriction: ");
        fgets(restriction, MAX_DIETARY_RESTRICTIONS, stdin);
        restriction[strcspn(restriction, "\n")] = '\0';
        addDietaryRestriction(restrictions, &restrictionCount, customerName, restriction);
        printf("Dietary restriction added.\n");
    } else if (choice == 8) {
        displayEvents(events, eventCount);
    } else if (choice == 9) {
        printf("Enter event name: ");
        fgets(eventName, MAX_NAME_LENGTH, stdin);
        eventName[strcspn(eventName, "\n")] = '\0';
        printf("Enter feedback: ");
        fgets(feedback, MAX_FEEDBACK, stdin);
        feedback[strcspn(feedback, "\n")] = '\0';
        printf("Enter rating (1-5): ");
        scanf("%d", &rating);
        getchar();
        collectFeedback(feedbacks, &feedbackCount, eventName, feedback, rating);
        printf("Feedback collected.\n");
    } else if (choice == 10) {
        saveEvents(events, eventCount);
        printf("Exiting...\n");
        exit(0);
    } else {
        printf("Invalid choice. Please try again.\n");
    }
}
```

```
    return 0;
}
void addMenuItem(MenuItem *menu, int *size, const char *itemName, float price) {
    strcpy(menu[*size].itemName, itemName);
    menu[*size].price = price;
    (*size)++;
}

void createEvent(Event *event, const char *eventName, const char *eventDate) {
    strcpy(event->eventName, eventName);
    strcpy(event->eventDate, eventDate);
    event->menuSize = 0;
}

void addMenuItemToEvent(Event *event, const char *itemName) {
    for (int i = 0; i < menuSize; i++) {
        if (strcmp(menu[i].itemName, itemName) == 0) {
            addMenuItem(event->menu, &event->menuSize, itemName, menu[i].price);
            return;
        }
    }
    printf("Menu item not found.\n");
}

void addOrder(Billing *billing, const char *customerName, const char *eventName, float amount) {
    strcpy(billing->orders[billing->orderCount].customerName, customerName);
    strcpy(billing->orders[billing->orderCount].eventName, eventName);
    billing->orders[billing->orderCount].totalAmount = amount;
    billing->orderCount++;
}
```

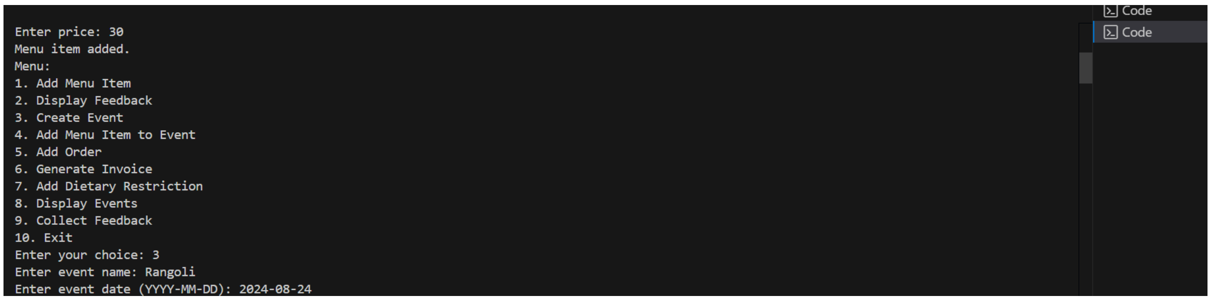
```
void generateInvoice(const Billing *billing) {
   for (int i = 0; i < billing->orderCount; i++) {
      printf("Invoice for %s\n", billing->orders[i].customerName);
      printf("Event: %s\n", billing->orders[i].eventName);
      printf("Total Amount: $%.2f\n", billing->orders[i].totalAmount);
      printf("-----------\n");
   }
}

void collectFeedback(Feedback *feedbacks, int *feedbackCount, const char *eventName, const char
*feedback, int rating) {
   strcpy(feedbacks[*feedbackCount].eventName, eventName);
   strcpy(feedbacks[*feedbackCount].feedback, feedback);
   feedbacks[*feedbackCount].rating = rating;
   (*feedbackCount)++;
}

void displayFeedback(const Feedback *feedbacks, int feedbackCount) {
   for (int i = 0; i < feedbackCount; i++) {
      printf("Feedback for %s:\n", feedbacks[i].eventName);
      printf("Rating: %d/5\n", feedbacks[i].rating);
      printf("%s\n", feedbacks[i].feedback);
      printf("-----------\n");
   }
}

void addDietaryRestriction(DietaryRestriction *restrictions, int *restrictionCount, const char
*customerName, const char *restriction) {
   strcpy(restrictions[*restrictionCount].customerName, customerName);
   strcpy(restrictions[*restrictionCount].restrictions, restriction);
   (*restrictionCount)++;
```

```
}

void displayEvents(const Event *events, int eventCount) {
    for (int i = 0; i < eventCount; i++) {
        printf("Event Name: %s\n", events[i].eventName);
        printf("Event Date: %s\n", events[i].eventDate);
        printf("Menu Items:\n");
        for (int j = 0; j < events[i].menuSize; j++) {
            printf("  %s: $%.2f\n", events[i].menu[j].itemName, events[i].menu[j].price);
        }
        printf("-----------\n");
    }
}

void saveEvents(const Event *events, int eventCount) {
    FILE *file = fopen("events.dat", "wb");
    if (file == NULL) {
        perror("Failed to open file");
        return;
    }
    fwrite(&eventCount, sizeof(int), 1, file);
    fwrite(events, sizeof(Event), eventCount, file);
    fclose(file);
}

void loadEvents(Event *events, int *eventCount) {
    FILE *file = fopen("events.dat", "rb");
    if (file == NULL) {
        perror("Failed to open file");
        return;
    }
    fread(eventCount, sizeof(int), 1, file);

    fread(events, sizeof(Event), *eventCount, file);

    fclose(file);

}


void displayMenu() {

    printf("Menu:\n");

    printf("1. Add Menu Item\n");

    printf("2. Display Feedback\n");

    printf("3. Create Event\n");

    printf("4. Add Menu Item to Event\n");

    printf("5. Add Order\n");

    printf("6. Generate Invoice\n");

    printf("7. Add Dietary Restriction\n");

    printf("8. Display Events\n");

    printf("9. Collect Feedback\n");

    printf("10. Exit\n");

    printf("Enter your choice: ");

}
```

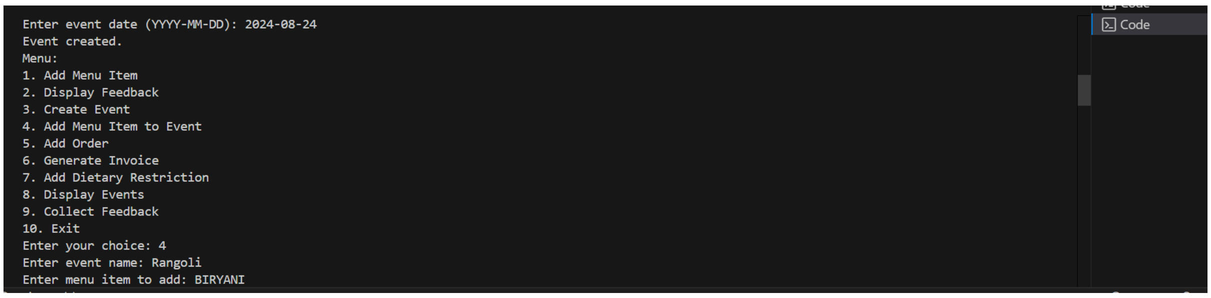
OUTPUT:



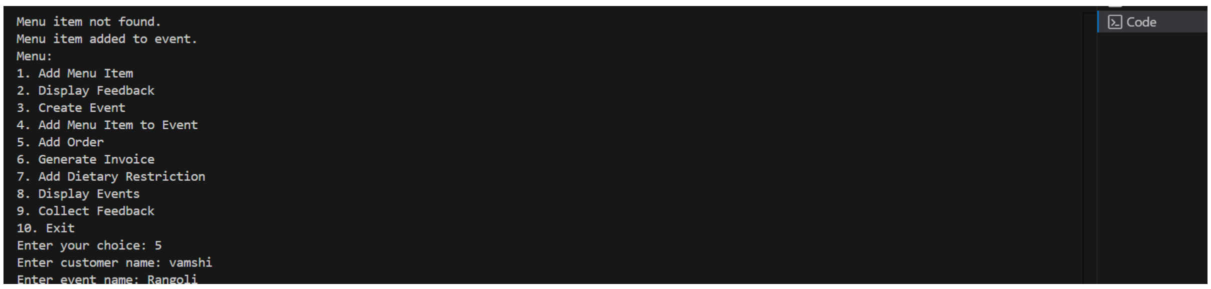
```
PS C:\Users\HP\AppData\Local\Temp> cd "C:\Users\HP\AppData\Local\Temp\" ; if ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($
?) { .\tempCodeRunnerFile }
Failed to open file: No such file or directory
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 1
Enter item name: DOSA
```



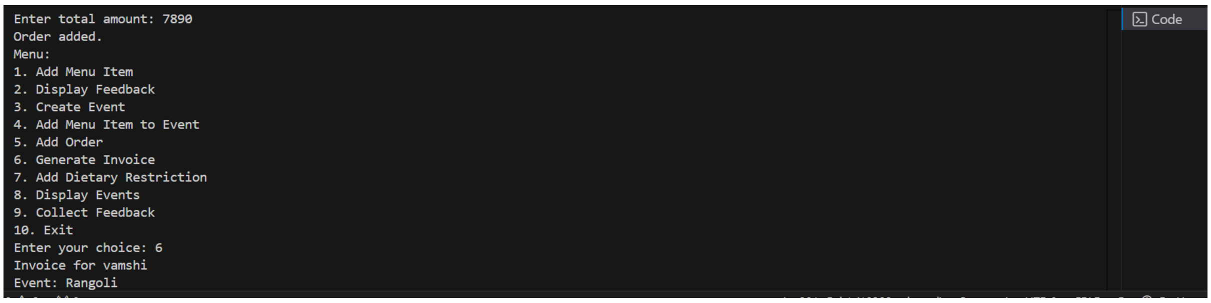
```
Enter price: 30
Menu item added.
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 3
Enter event name: Rangoli
Enter event date (YYYY-MM-DD): 2024-08-24
```



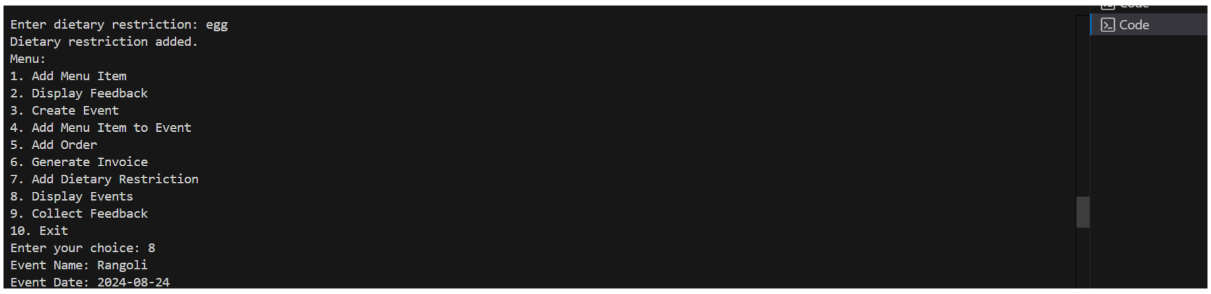
```
Enter event date (YYYY-MM-DD): 2024-08-24
Event created.
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 4
Enter event name: Rangoli
Enter menu item to add: BIRYANI
```



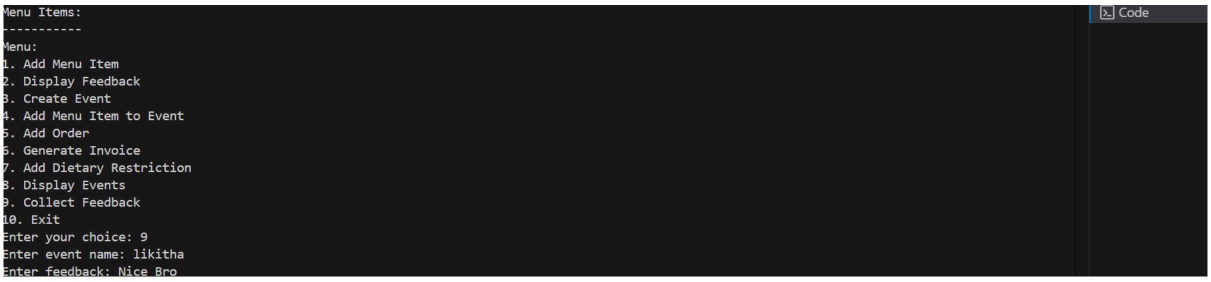
```
Menu item not found.
Menu item added to event.
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 5
Enter customer name: vamshi
Enter event name: Rangoli
```



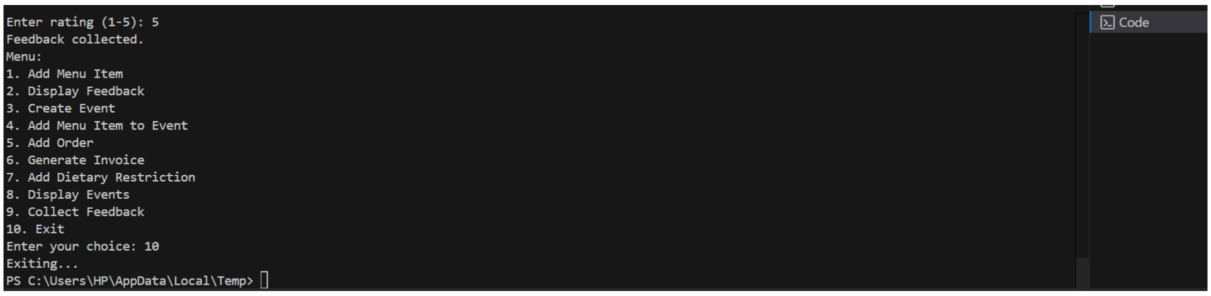
```
Enter total amount: 7890
Order added.
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 6
Invoice for vamshi
Event: Rangoli
```



```
Enter dietary restriction: egg
Dietary restriction added.
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 8
Event Name: Rangoli
Event Date: 2024-08-24
```



```
Menu Items:
----------
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 9
Enter event name: likitha
Enter feedback: Nice Bro
```



```
Enter rating (1-5): 5
Feedback collected.
Menu:
1. Add Menu Item
2. Display Feedback
3. Create Event
4. Add Menu Item to Event
5. Add Order
6. Generate Invoice
7. Add Dietary Restriction
8. Display Events
9. Collect Feedback
10. Exit
Enter your choice: 10
Exiting...
PS C:\Users\HP\AppData\Local\Temp>
```

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <stdbool.h>


#define MAX_DONORS 1000

#define MAX_CAMPAIGNS 1000


int donor_count = 0;

int campaign_count = 0;


typedef struct {

   int id;

   char name[50];

   int age;

   long long phone;

   char email[100];

   int pincode;

   char address[200];

   float amount;

} Donor;


typedef struct {
    int id;
    char name[50];
    float funds;
} Campaign;

Donor *donors;
Campaign *campaigns;

void send_thank_you_email(int id, const char *email);
void allocate_funds(int id, float amount);
void allocate_fund_to_campaign();
void create_campaign();
void impact_report();
void print_all_donors();
void get_donor_details();
void track_donations();
int register_donor(int id, const char *name, int age, long long phone, const char *email, int pincode,
const char *address);
void track_donation_by_phone(long long phone);
void track_donation_by_id(int id);

void send_thank_you_email(int id, const char *email) {
    printf("Sending thank-you email to donor ID: %d at %s\n", id, email);
}

void allocate_funds(int id, float amount) {
    bool success = false;
    char email[100];

    for (int i = 0; i < donor_count; ++i) {
        if (id == donors[i].id) {
            donors[i].amount += amount;
            strcpy(email, donors[i].email);
            success = true;
            break;
        }
    }

    if (success) {
        send_thank_you_email(id, email);
    } else {
        printf("Allocation failed. No donor found with ID %d.\n", id);
    }
}

void generate_receipt(int id) {
    for (int i = 0; i < donor_count; ++i) {
        if (id == donors[i].id) {
            printf("Receipt for Donor ID: %d\n", donors[i].id);
            printf("Name: %s\n", donors[i].name);
            printf("Age: %d\n", donors[i].age);
            printf("Phone: %lld\n", donors[i].phone);
            printf("Email: %s\n", donors[i].email);
            printf("Pincode: %d\n", donors[i].pincode);
            printf("Address: %s\n", donors[i].address);
            printf("Amount Donated: $%.2f\n", donors[i].amount);
            printf("-----------------------------\n");
            return;
        }
    }
    printf("No donor found with ID %d.\n", id);
}
```

```
int register_donor(int id, const char *name, int age, long long phone, const char *email, int pincode,
const char *address) {
    if (donor_count >= MAX_DONORS) {
        printf("Maximum number of donors reached.\n");
        return 0;
    }
    for (int i = 0; i < donor_count; ++i) {
        if (donors[i].id == id) {
            printf("Donor ID already exists.\n");
            return 0;
        }
    }

    Donor *new_donor = &donors[donor_count];
    new_donor->id = id;
    strncpy(new_donor->name, name, sizeof(new_donor->name) - 1);
    new_donor->name[sizeof(new_donor->name) - 1] = '\0';
    new_donor->age = age;
    new_donor->phone = phone;
    strncpy(new_donor->email, email, sizeof(new_donor->email) - 1);
    new_donor->email[sizeof(new_donor->email) - 1] = '\0';
    new_donor->pincode = pincode;
    strncpy(new_donor->address, address, sizeof(new_donor->address) - 1);
    new_donor->address[sizeof(new_donor->address) - 1] = '\0';
    new_donor->amount = 0;

    donor_count++;

    return 1;
}
```

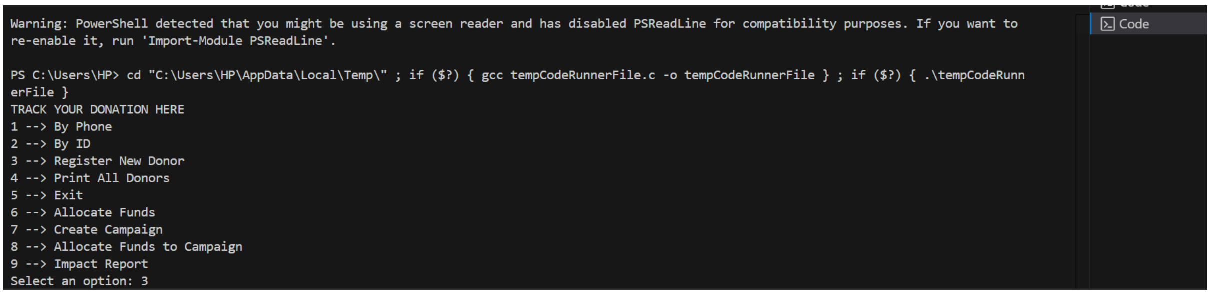
```
void track_donation_by_phone(long long phone) {
    for (int i = 0; i < donor_count; ++i) {
        if (donors[i].phone == phone) {
            printf("Donor found:\n");
            printf("ID: %d\n", donors[i].id);
            printf("Name: %s\n", donors[i].name);
            printf("Age: %d\n", donors[i].age);
            printf("Phone: %lld\n", donors[i].phone);
            printf("Email: %s\n", donors[i].email);
            printf("Pincode: %d\n", donors[i].pincode);
            printf("Address: %s\n", donors[i].address);
            printf("Amount Donated: $%.2f\n", donors[i].amount);
            return;
        }
    }
    printf("No donor found with that phone number.\n");
}

void track_donation_by_id(int id) {
    for (int i = 0; i < donor_count; ++i) {
        if (donors[i].id == id) {
            printf("Donor found:\n");
            printf("ID: %d\n", donors[i].id);
            printf("Name: %s\n", donors[i].name);
            printf("Age: %d\n", donors[i].age);
            printf("Phone: %lld\n", donors[i].phone);
            printf("Email: %s\n", donors[i].email);
            printf("Pincode: %d\n", donors[i].pincode);
            printf("Address: %s\n", donors[i].address);
            printf("Amount Donated: $%.2f\n", donors[i].amount);
            return;
        }
    }
    printf("No donor found with that ID.\n");
}

void print_all_donors() {
    for (int i = 0; i < donor_count; ++i) {
        printf("Donor ID: %d\n", donors[i].id);
        printf("Name: %s\n", donors[i].name);
        printf("Age: %d\n", donors[i].age);
        printf("Phone: %lld\n", donors[i].phone);
        printf("Email: %s\n", donors[i].email);
        printf("Pincode: %d\n", donors[i].pincode);
        printf("Address: %s\n", donors[i].address);
        printf("Amount Donated: $%.2f\n", donors[i].amount);
        printf("-----------------------------\n");
    }
}

void get_donor_details() {
    int id;
    char name[50];
    int age;
    long long phone;
    char email[100];
    int pincode;
    char address[200];
    float amount;

    printf("Enter donor ID: ");
    scanf("%d", &id);

    printf("Enter donor name: ");
    scanf(" %[^\n]s", name);

    printf("Enter donor age: ");
    scanf("%d", &age);

    printf("Enter donor phone: ");
    scanf("%lld", &phone);

    printf("Enter donor email: ");
    scanf(" %[^\n]s", email);

    printf("Enter donor pincode: ");
    scanf("%d", &pincode);

    printf("Enter donor address: ");
    scanf(" %[^\n]s", address);

    printf("Enter amount donated: ");
    scanf("%f", &amount);

    int result = register_donor(id, name, age, phone, email, pincode, address);
    if(result == 1) {
        allocate_funds(id, amount);
        generate_receipt(id);
    } else {
        printf("Registration failed. Please try again.\n");
    }
}
```

```c
void allocate_fund_to_campaign() {
    int campaign_id;
    float amount;
    printf("Enter campaign ID: ");
    scanf("%d", &campaign_id);

    Campaign *campaign = NULL;
    for (int i = 0; i < campaign_count; ++i) {
        if (campaigns[i].id == campaign_id) {
            campaign = &campaigns[i];
            break;
        }
    }

    if (campaign == NULL) {
        printf("No campaign found with ID %d.\n", campaign_id);
        return;
    }

    printf("Enter amount to allocate: ");
    scanf("%f", &amount);

    campaign->funds += amount;
    printf("Allocated $%.2f to Campaign ID %d.\n", amount, campaign_id);
}

void create_campaign() {
    if (campaign_count >= MAX_CAMPAIGNS) {
        printf("Maximum number of campaigns reached.\n");
        return;
    }

    int id;
    char name[50];

    printf("Enter campaign ID: ");
    scanf("%d", &id);
    for (int i = 0; i < campaign_count; ++i) {
        if (campaigns[i].id == id) {
            printf("Campaign ID already exists.\n");
            return;
        }
    }

    printf("Enter campaign name: ");
    scanf(" %[^\n]s", name);

    Campaign *new_campaign = &campaigns[campaign_count];
    new_campaign->id = id;
    strncpy(new_campaign->name, name, sizeof(new_campaign->name) - 1);
    new_campaign->name[sizeof(new_campaign->name) - 1] = '\0';
    new_campaign->funds = 0;
    campaign_count++;

    printf("Campaign created successfully.\n");
}

void impact_report() {
    printf("Impact Report:\n");
    for (int i = 0; i < campaign_count; ++i) {
        printf("Campaign ID: %d\n", campaigns[i].id);
        printf("Campaign Name: %s\n", campaigns[i].name);
        printf("Funds Allocated: $%.2f\n", campaigns[i].funds);
        printf("-----------------------------\n");
    }
}

void track_donations() {
    int option;
    printf("TRACK YOUR DONATION HERE\n");
    printf("1 --> By Phone\n");
    printf("2 --> By ID\n");
    printf("3 --> Register New Donor\n");
    printf("4 --> Print All Donors\n");
    printf("5 --> Exit\n");
    printf("6 --> Allocate Funds\n");
    printf("7 --> Create Campaign\n");
    printf("8 --> Allocate Funds to Campaign\n");
    printf("9 --> Impact Report\n");
    printf("Select an option: ");
    scanf("%d", &option);

    switch (option) {
        case 1: {
            long long phone;
            printf("Enter phone number: ");
            scanf("%lld", &phone);
            track_donation_by_phone(phone);
            break;
        }
        case 2: {
            int id;
            printf("Enter ID: ");
            scanf("%d", &id);
            track_donation_by_id(id);
            break;
        }
        case 3:
            get_donor_details();
            break;
        case 4:
            print_all_donors();
            break;
        case 5:
            printf("Exiting...\n");
            free(donors);
            free(campaigns);
            exit(0);
            break;
        case 6: {
            int id;
            float amount;
            printf("Enter ID and amount: ");
            scanf("%d %f", &id, &amount);
            allocate_funds(id, amount);
            break;
        }
        case 7:
            create_campaign();
            break;
        case 8:
            allocate_fund_to_campaign();
            break;
```

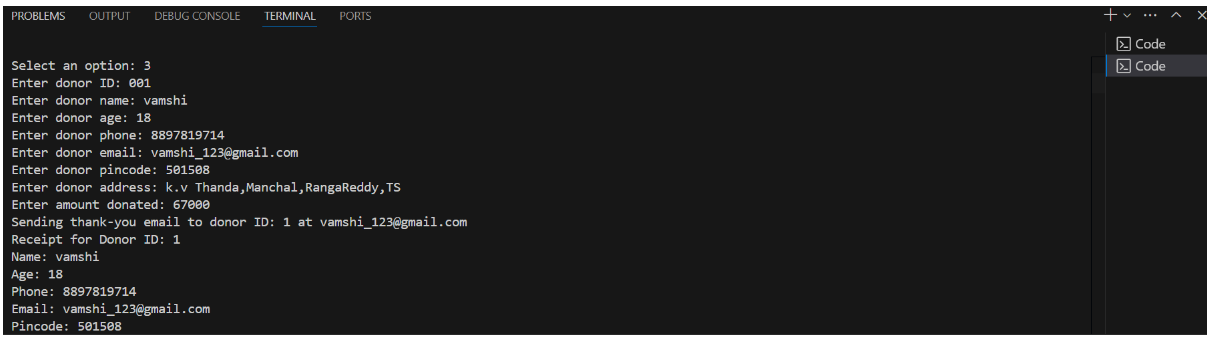
```
        case 9:
            impact_report();
            break;
        default:
            printf("Invalid option. Try again.\n");
            break;
    }
}

int main() {
    donors = malloc(MAX_DONORS * sizeof(Donor));
    if (donors == NULL) {
        printf("Memory allocation failed.\n");
        return 1;
    }

    campaigns = malloc(MAX_CAMPAIGNS * sizeof(Campaign));
    if (campaigns == NULL) {
        printf("Memory allocation failed.\n");
        free(donors);
        return 1;
    }

    while (1) {
        track_donations();
    }

    free(donors);
    free(campaigns);

    return 0;
```

}

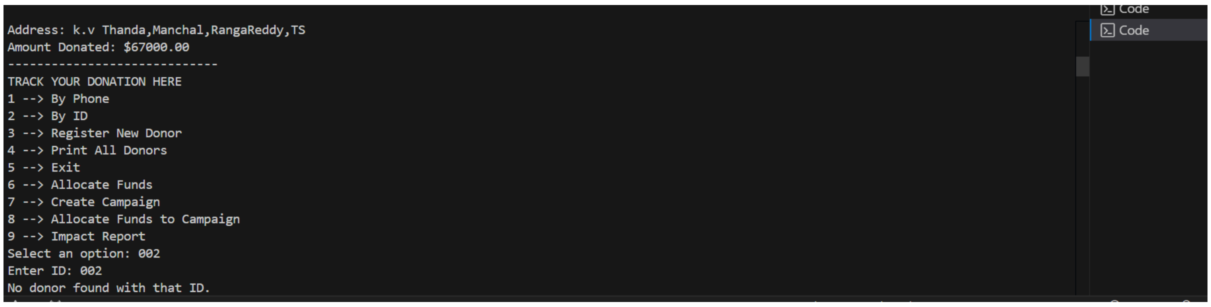
OUTPUT:



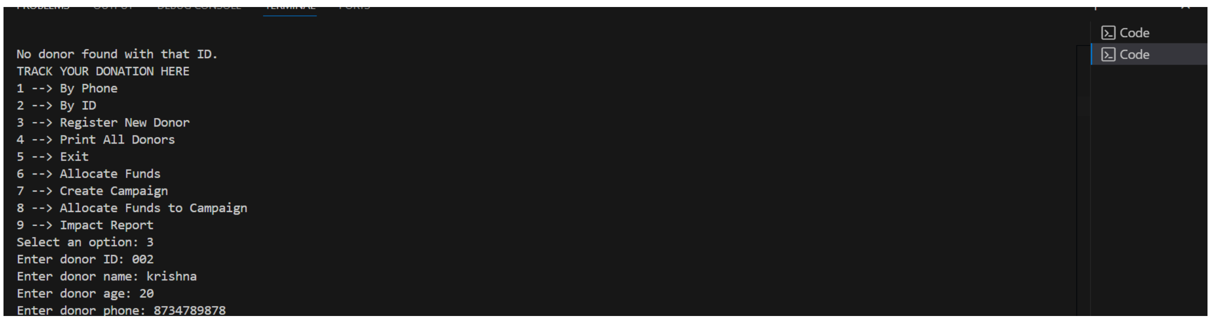
```
Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadLine for compatibility purposes. If you want to
re-enable it, run 'Import-Module PSReadLine'.

PS C:\Users\HP> cd "C:\Users\HP\AppData\Local\Temp\" ; if ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunn
erFile }
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 3
```



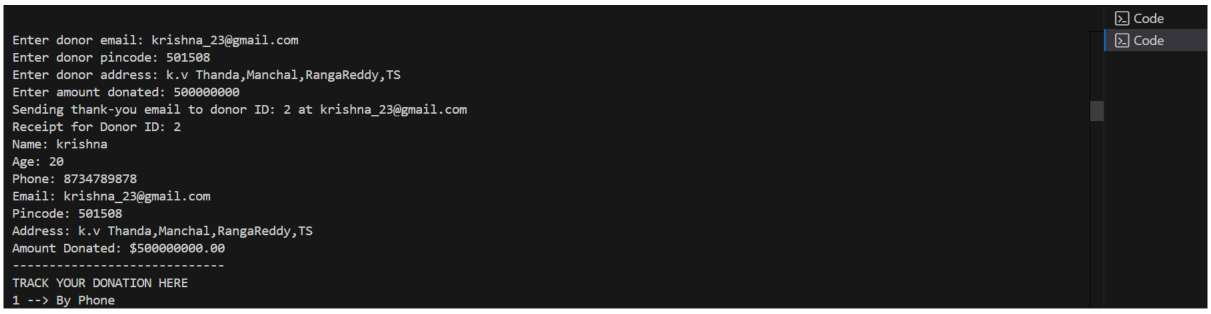
```
Select an option: 3
Enter donor ID: 001
Enter donor name: vamshi
Enter donor age: 18
Enter donor phone: 8897819714
Enter donor email: vamshi_123@gmail.com
Enter donor pincode: 501508
Enter donor address: k.v Thanda,Manchal,RangaReddy,TS
Enter amount donated: 67000
Sending thank-you email to donor ID: 1 at vamshi_123@gmail.com
Receipt for Donor ID: 1
Name: vamshi
Age: 18
Phone: 8897819714
Email: vamshi_123@gmail.com
Pincode: 501508
```



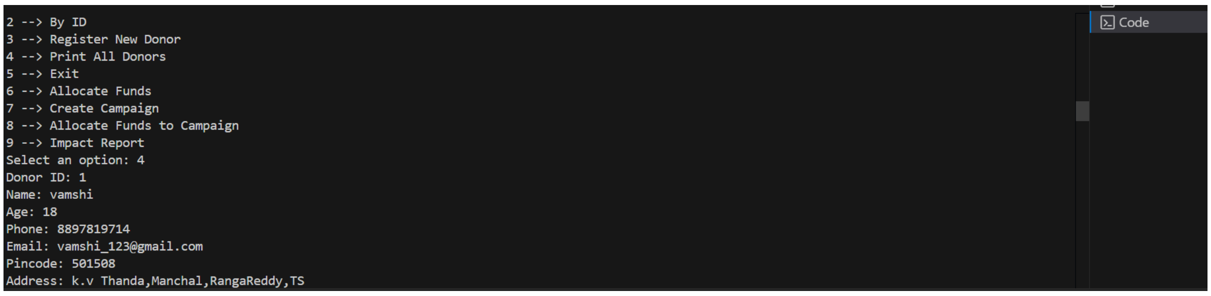
```
Address: k.v Thanda,Manchal,RangaReddy,TS
Amount Donated: $67000.00
----------------------------
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 002
Enter ID: 002
No donor found with that ID.
```



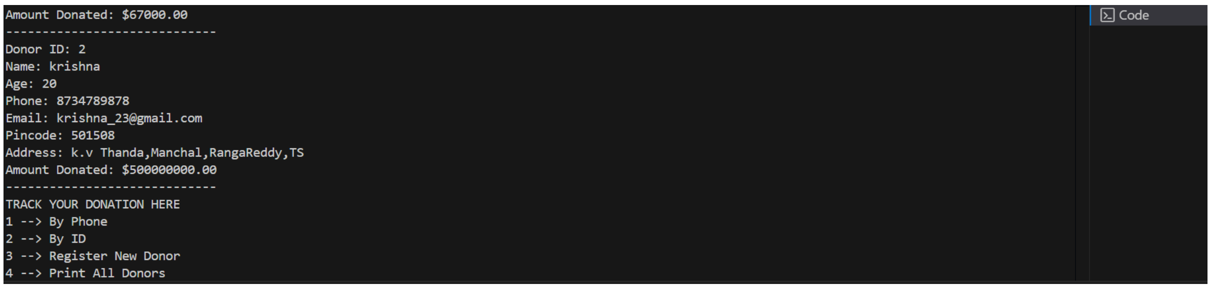
```
No donor found with that ID.
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 3
Enter donor ID: 002
Enter donor name: krishna
Enter donor age: 20
Enter donor phone: 8734789878
```



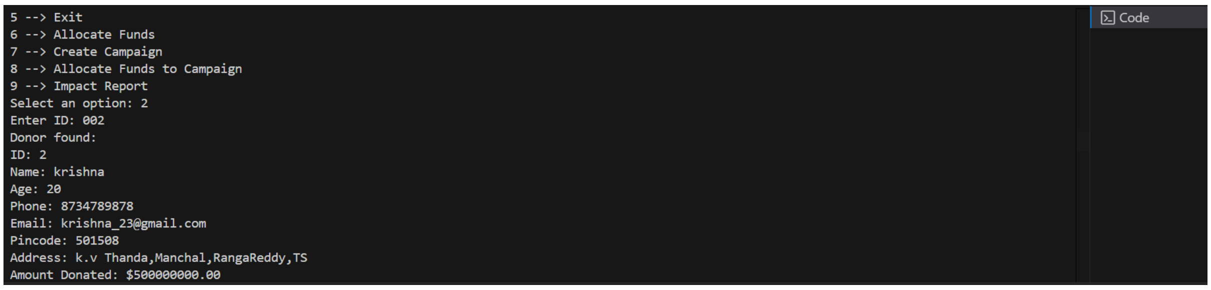
```
Enter donor email: krishna_23@gmail.com
Enter donor pincode: 501508
Enter donor address: k.v Thanda,Manchal,RangaReddy,TS
Enter amount donated: 500000000
Sending thank-you email to donor ID: 2 at krishna_23@gmail.com
Receipt for Donor ID: 2
Name: krishna
Age: 20
Phone: 8734789878
Email: krishna_23@gmail.com
Pincode: 501508
Address: k.v Thanda,Manchal,RangaReddy,TS
Amount Donated: $500000000.00
----------------------------
TRACK YOUR DONATION HERE
1 --> By Phone
```



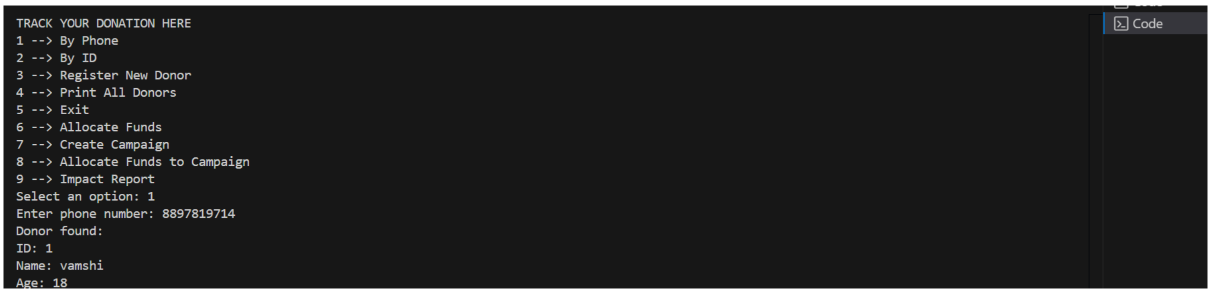
```
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 4
Donor ID: 1
Name: vamshi
Age: 18
Phone: 8897819714
Email: vamshi_123@gmail.com
Pincode: 501508
Address: k.v Thanda,Manchal,RangaReddy,TS
```



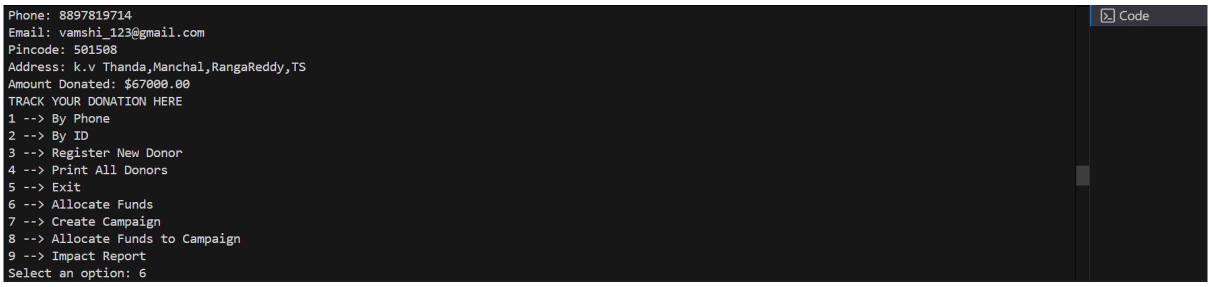
```
Amount Donated: $67000.00
----------------------------
Donor ID: 2
Name: krishna
Age: 20
Phone: 8734789878
Email: krishna_23@gmail.com
Pincode: 501508
Address: k.v Thanda,Manchal,RangaReddy,TS
Amount Donated: $500000000.00
----------------------------
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
```



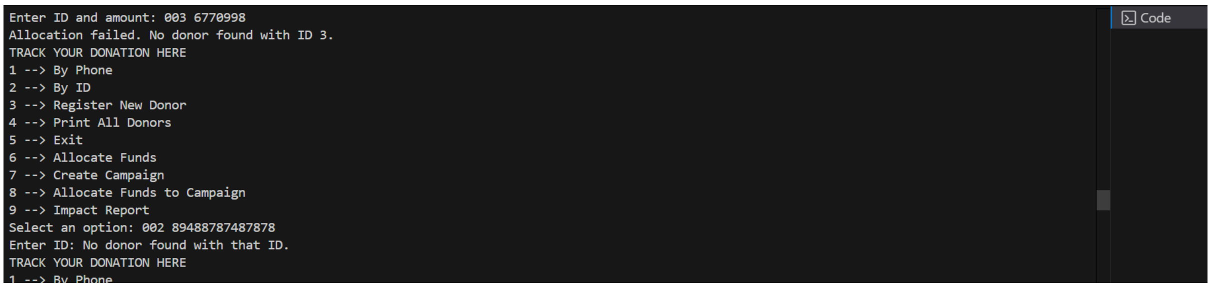
```
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 2
Enter ID: 002
Donor found:
ID: 2
Name: krishna
Age: 20
Phone: 8734789878
Email: krishna_23@gmail.com
Pincode: 501508
Address: k.v Thanda,Manchal,RangaReddy,TS
Amount Donated: $500000000.00
```



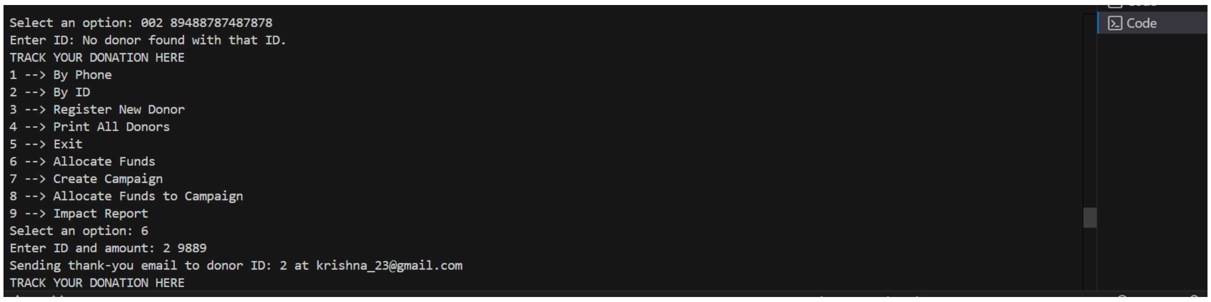
```
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 1
Enter phone number: 8897819714
Donor found:
ID: 1
Name: vamshi
Age: 18
```



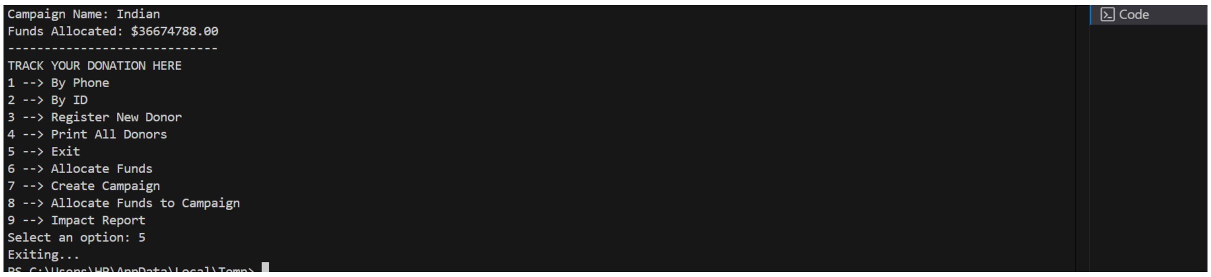
```
Phone: 8897819714
Email: vamshi_123@gmail.com
Pincode: 501508
Address: k.v Thanda,Manchal,RangaReddy,TS
Amount Donated: $67000.00
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 6
```



```
Enter ID and amount: 003 6770998
Allocation failed. No donor found with ID 3.
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 002 89488787487878
Enter ID: No donor found with that ID.
TRACK YOUR DONATION HERE
1 --> By Phone
```



```
Select an option: 002 89488787487878
Enter ID: No donor found with that ID.
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 6
Enter ID and amount: 2 9889
Sending thank-you email to donor ID: 2 at krishna_23@gmail.com
TRACK YOUR DONATION HERE
```



```
Campaign Name: Indian
Funds Allocated: $36674788.00
----------------------------
TRACK YOUR DONATION HERE
1 --> By Phone
2 --> By ID
3 --> Register New Donor
4 --> Print All Donors
5 --> Exit
6 --> Allocate Funds
7 --> Create Campaign
8 --> Allocate Funds to Campaign
9 --> Impact Report
Select an option: 5
Exiting...
PS C:\Users\HP\AppData\Local\Temp>
```