NAME: PRASHANT UPPAR

SRN: 02FE22BCS069

ROLL NO:21

TEAM NO:08

CODE

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

    int i = 0, j = 0, k = 0, a, b;

    char location[50];

    int adj[14][14];

    // Function to print lines for formatting purpose

    int user_signup()
{
    FILE *fp=fopen("user_info.txt","r+");

     if (fp == NULL)
       {
           printf("\n File is not found");

           return 1;
```

```
        }

    char username[50],password[50];

    printf("\n Enter the user name:-\t");

    scanf("%s",username);

    printf("\n Enter the password:-\t");

    scanf("%s",password);

    fprintf(fp,"%s %s",username,password);

    printf("\n Your account is successfully created.......");

    fclose(fp);


}
    void print_line()
    {
        printf("\n\n------------------------------------------------------------\n\n");
    }



    // Function to print pattern  for the formatting purpose
    void print_pattern()
    {
        printf("\n\n::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::\n\n");
    }
```

```
// structure declaration for storing the cities name read through files

typedef struct cities

{

   int city_id;

   char city_name[50];

} CT;


CT c[1000];



// structure declaration for storing the venue halls name read through files

typedef struct venue

{

   int venue_id;

   char venue_hall[50];

   int price;

} ve;


ve v[20];



// structure declaration for storing the events name read through files

typedef struct events

{

   int Id;

   char event_name[30];

} Et;


Et e[50];
```

```
//// structure declaration for storing the user information  read through files
typedef struct user_info {
   char username[50];
   char password[50];
} User;

User users[10];


// structure declaration for storing the date of event booked and user id  read through user
typedef struct venue_bookings
{
   int date,user_id;
}month;

month m[25];



struct set
{
  int key;
  int data;
};



struct set *array;

// number of cells in the months
```

```
int capacity = 31;


// number of booking took place

int size = 0;




// hash function to calculate hashing address on given key given by the user

int hashFunction(int key)

{

  return (key % capacity);

}


// Checking the user login credentials are present in the domain

int authenticate_user()

{


    char username[50], password[50];


    printf("\nEnter your username: ");


    scanf("%s", username);


    printf("Enter your password: ");


    scanf("%s", password);


    FILE *userFile = fopen("user_info.txt","r");


    if (userFile == NULL)

        {

        printf("Error opening user_info.txt\n");
```

```
   exit(1);
   }

int validUser = 0;

while (fscanf(userFile, "%s %s", users[i].username, users[i].password) == 2)
   {

   if (strcmp(username, users[i].username) == 0 && strcmp(password, users[i].password) == 0) {

      validUser = 1;

      break;
   }
}

fclose(userFile);

if (validUser)
   {
   printf("\nAuthentication successful. Proceeding to the program.\n");

   return 1;

   }
else
   {
     printf("\nInvalid username or password. Exiting program.\n");

     return 0;
```

```
        }
}

// Function to delete the event given to the manager
void delete_event(int event_id)
{
    FILE *eventFile = fopen("event1.txt", "r");

    if (eventFile == NULL)
        {
        printf("Error opening event1.txt\n");
        exit(1);
        }

    FILE *tempFile = fopen("temp_event1.txt", "w");

    if (tempFile == NULL) {
        printf("Error opening temp_event1.txt\n");
        exit(1);
    }

    while (fscanf(eventFile, "%d %s", &e[i].Id, e[i].event_name) == 2) {
        if (e[i].Id != event_id) {
            fprintf(tempFile, "%d %s\n", e[i].Id, e[i].event_name);
        }
    }

    fclose(eventFile);
    fclose(tempFile);
    remove("event1.txt");
    rename("temp_event1.txt", "event1.txt");

    printf("\nEvent with ID %d has been deleted.\n", event_id);
}

// Function to delete venue hall based on given id
void delete_venue(int venue_id) {
    FILE *venueFile = fopen("venues.txt", "r");

    if (venueFile == NULL) {
        printf("Error opening venues.txt\n");
        exit(1);
    }

    FILE *tempFile = fopen("temp_venues.txt", "w");

    if (tempFile == NULL) {
        printf("Error opening temp_venues.txt\n");
        exit(1);
    }

    while (fscanf(venueFile, "%d %s %d", &v[j].venue_id, v[j].venue_hall, &v[j].price) == 3) {
        if (v[j].venue_id != venue_id) {
            fprintf(tempFile, "%d %s %d\n", v[j].venue_id, v[j].venue_hall, v[j].price);
        }
    }

    fclose(venueFile);
    fclose(tempFile);
    remove("venues.txt");
    rename("temp_venues.txt", "venues.txt");


    printf("\nVenue with ID %d has been deleted.\n", venue_id);
}



// functtion find prime used in hash function
int checkPrime(int n)
{
  int i;
  if (n == 1 || n == 0)
  {
    return 0;
  }
  for (i = 2; i < n / 2; i++)
  {
    if (n % i == 0)
    {
      return 0;
    }
  }
  return 1;
}

// checking prime or not
int getPrime(int n)
{
  if (n % 2 == 0)
  {
    n++;
  }
  while (!checkPrime(n))
  {
    n += 2;
  }
  return n;
}


void init_array()
{
  capacity = getPrime(capacity);
  array = (struct set *)malloc(capacity * sizeof(struct set));
  for (int i = 0; i < capacity; i++)
  {
    array[i].key = 0;
    array[i].data = 0;
  }
}

void insert(int key, int data)
{
  int index = hashFunction(key);


  FILE *file = fopen("hash_table_data.txt", "a");
  if (file == NULL)
  {
    printf("Error opening file.\n");
    exit(1);
  }
```

```
FILE *readFile = fopen("hash_table_data.txt", "r");
if (readFile != NULL)
{
  int fileKey;
  while (fscanf(readFile, "%d", &fileKey) != EOF)
  {
    if (fileKey == key)
    {
      printf("\n date (%d) already booked by others.\n", key);
      fclose(readFile);
      fclose(file);
      return;
    }
  }
  fclose(readFile);
}


fprintf(file, "%d %d\n", key, data);
fclose(file);

if (array[index].data == 0)
{
  array[index].key = key;
  array[index].data = data;
  size++;
  printf("\n date  (%d) has been booked successfully  \n", key);
}
else if (array[index].key == key)
```

```
 {
   array[index].data = data;
 }
 else
 {
   printf("\n collision occured date already booked \n");
 }
}

// for the undo booking
void remove_element(int key)
{
 int index = hashFunction(key);
 if (array[index].data == 0)
 {
   printf("\n This date key  does not exist \n");
 }
 else
 {
   array[index].key = 0;
   array[index].data = 0;
   size--;
   printf("\n Key (%d) has been removed \n", key);
 }


 FILE *file = fopen("hash_table_data.txt", "r");
 if (file == NULL)
 {
   printf("Error opening file.\n");
   exit(1);
```

```
    }

    FILE *tempFile = fopen("temp_hash_table_data.txt", "w");
    if (tempFile == NULL)
    {
      printf("Error opening temp file.\n");
      exit(1);
    }

    int fileKey, fileData;
    while (fscanf(file, "%d %d", &fileKey, &fileData) != EOF)
    {
      if (fileKey != key)
      {
        fprintf(tempFile, "%d %d\n", fileKey, fileData);
      }
    }

    fclose(file);
    fclose(tempFile);

    remove("hash_table_data.txt");
    rename("temp_hash_table_data.txt", "hash_table_data.txt");
  }

  // to display booking details
  void display()
```

```
{
  int i;
  for (i = 0; i < capacity; i++)
  {
    if (array[i].data == 0)
    {
      printf("\n array[%d]: / ", i);
    }
    else
    {
      printf("\n date: %d feb[%d]: %d \t", array[i].key, i, array[i].data);
    }
  }
}

int size_of_hashtable()
{
  return size;
}



// function to read number of rows in the files where adjacency matrix is stored
int calorder()
{
    FILE *fp3 = fopen("adjacency_matrix.txt", "r");

    char ch;
    int row = 0;

    if (fp3 == NULL)
    {
        printf("\n cannot open the file");
        return 1;
    }

    while ((ch = fgetc(fp3)) != '\n')
    {
        if (ch == ',')
        {
            row++;
        }
    }

    return row;

    fclose(fp3);
}


// Load the adjacency matrix into adjcency matric from the file called adjacency matrix
void load_adjacency()
{
    FILE *fp4;
    fp4 = fopen("adjacency_matrix.txt", "r");
    int n = calorder();
    int temp;
    char s;

    for (i = 0; i < 14; i++)
    {
        for (j = 0; j < 14; j++)
        {
            fscanf(fp4, "%d%c", &temp, &s);
            adj[i][j] = temp;
        }
    }

    fclose(fp4);

}

// function calculate min distance between two areas where is it min than updated
int minDistance(int dist[], int sptSet[])
{
    int min = INT_MAX, min_index;

    for (int v = 0; v < k; v++)
    {
        if (sptSet[v] == 0 && dist[v] <= min)
        {
            min = dist[v];
            min_index = v;
        }
    }

    return min_index;
}



// printing the shortest path
void printSolution(int dist[], int n, int parent[])
{
    printf("\n Node\tDistance\tPath\n");
    for (int i = 0; i < k; i++)
    {
        printf("%d\t%d\t\t%d", i, dist[i], i);

        int j = i;
        while (parent[j] != -1)
        {
            printf(" <- %d", parent[j]);
            j = parent[j];
        }

        printf("\n");
    }
}

// Dijkstra's algorithm of finding shortest path
void dijkstra(int src, int dest)
{

    int dist[k];
    int parent[k];



    int sptSet[k];



    for (int i = 0; i < k; i++)
    {
        dist[i] = INT_MAX;
        sptSet[i] = 0;
        parent[i] = -1;
    }



    dist[src] = 0;



    for (int count = 0; count < k - 1; count++)
    {

        int u = minDistance(dist, sptSet);

        sptSet[u] = 1;

        for (int v = 0; v < k; v++)
        {
            if (!sptSet[v] && adj[u][v] && dist[u] != INT_MAX &&
                dist[u] + adj[u][v] < dist[v])
            {
                dist[v] = dist[u] + adj[u][v];

                parent[v] = u;
            }
        }
    }



    printSolution(dist, k, parent);

    printf("\n\n Shortest Path from %s to %s: %d\n", c[src].city_name, c[dest].city_name, dist[dest]);
}
```

```
// searching a city is is present in the considered domain
int searchCity(char pattern[])
{
   int N;
   for (N = 0; N < 16; N++)
      ;

   int M = 3;

   for (int i = 0; i <= N - M; i++)
   {
      int j;
      for (j = 0; j < M; j++)
      {
         if (c[i].city_name[j] != pattern[j])
            break;
      }

      if (j == M)
      {
         return i;
      }
   }

   return -1;
}

void swap(ve *a, ve *b)
{
   ve temp = *a;
```

```
    *a = *b;

    *b = temp;

}


// Quick sort partition function

int partition(ve arr[], int low, int high)

{

    int pivot = arr[high].price;

    int i = (low - 1);


    for (int j = low; j <= high - 1; j++)

    {

        if (arr[j].price < pivot)

        {

            i++;

            swap(&arr[i], &arr[j]);

        }

    }

    swap(&arr[i + 1], &arr[high]);

    return (i + 1);

}


// Quick sort main function

void quickSort(ve arr[], int low, int high)

{

    if (low < high)

    {

        int pi = partition(arr, low, high);

        quickSort(arr, low, pi - 1);

        quickSort(arr, pi + 1, high);

    }
```

```
}



int select_event()
{
    printf("\n\n Enter the Id number of the event you want to choose: ");
    scanf("%d", &a);

    printf("\n Your events choice is successfully processed.......");
    return a;
}



// Function to load and display events to the user
int load_events()
{
    FILE *fp1;
    fp1 = fopen("event1.txt", "r");

    if (fp1 == NULL)
    {
        printf("\n File is not found");
        return 1;
    }

    print_pattern();

    printf("\n DISPLAYING EVENTS LIST >>>>>>>>>>>>>>>>>>>> ");

    print_line();
    printf("\n -:The events at UK 27 are :-");

    print_pattern();

    printf("\n EVENT ID | EVENT NAME | PRICE");

    printf("\n ------------------------------------------");



    while (fscanf(fp1, "%d %s", &e[i].Id, e[i].event_name) == 2)
    {
        printf("\n %d.%s", e[i].Id, e[i].event_name);
        i++;
    }




    fclose(fp1);

    a = select_event();

    return a;
}



// select the venue hall you want
int select_venue_hall()
{
    printf("\n\n Enter the id of venue you want to book:-");
    scanf("%d", &b);
    printf("\n Your venue hall choice is successfully processed.... ");

    return b;
}



// load and display venue halls to the user
int venue_halls()
{
    FILE *fp1 = fopen("venues.txt", "r");

    if (fp1 == NULL)
    {
        printf("\n File is not found");
        return 1;
    }

    print_pattern();

    printf("\n DISPLAYING VENUE HALLS LIST>>>>>>>>>>>>>>>>>>>");



    printf("\n\n\n -: The different venue hall at UK 27 are :-");



    print_pattern();

    printf("\n VENUE ID | VENUE HALL NAME | PRICE");

    printf("\n ---------------------------------------------------");
```

```
while (fscanf(fp1, "%d %s %d", &v[j].venue_id, v[j].venue_hall, &v[j].price) == 3)
{
   printf("\n %d.%s %d", v[j].venue_id, v[j].venue_hall, v[j].price);
   j++;
}



fclose(fp1);

int r;

printf("\n\n Press 1 if u want to get display of venue hall in order of cost:-");

scanf("%d",&r);

if(r==1)

{

printf("\n\n PROCESSING DATA >>>>>>>>>>>>>>>>>>>");


printf("\n\n The  venue hall in sorted order are :-%d", j);



printf("\n\n");

quickSort(v, 0, j - 1);
```

```
        printf("\n VENUE ID | VENUE HALL NAME | PRICE");

    printf("\n ---------------------------------------------------");

    for (int h = 0; h < j; h++)
    {
        printf("\n %d.%s %d", v[h].venue_id, v[h].venue_hall, v[h].price);
    }




    }
    b = select_venue_hall();

    return 1;
}



// function to check user area
int domain_check()
{
    char cityPattern[4];

    printf("\n Enter the first 3 characters of the city name: ");

    scanf("%s", cityPattern);

    int cityIndex = searchCity(cityPattern);
    if (cityIndex != -1)

    {

        printf("\n Area found: %s\n", c[cityIndex].city_name);

    }

    else

    {

        printf("\n City not found\n");

    }


    return 1;

}


// function to check user area

int travel_venue()

{

    FILE *fp2 = fopen("cities.txt", "r");


    if (fp2 == NULL)

    {

        printf("\n File is not found");

        return 1;

    }


    printf("\n DISPLAYING AREAS OF THE BELAGAVI CITY>>>>>>>>>>>>>>>>>>>");


    printf("\n -:The the areas of belagavi under consideration of the shortest path:-");



    printf("\n AREA ID | AREA NAME ");


    printf("\n-----------------------------------------------");
    k = 0;

    while (fscanf(fp2, "%d%s", &c[k].city_id, c[k].city_name) == 2)
    {
        printf("\n %d.%s", c[k].city_id, c[k].city_name);
        k++;
    }

    fclose(fp2);



    domain_check();

    return 1;
}



// user booking perticular date for venue
void book_venue()
{
int choice, key, data, n;
  int c = 0;
  init_array();

  do
  { print_pattern();
```

```
printf("\n1.Book the venue for the event"
    "\n2.Undo your booking"
    "\n3.Check number of booking took place in that month"
    "\n4.Display booking details"
    "\n\n Please enter your choice: ");

scanf("%d", &choice);
switch (choice)
{
case 1:

  printf("\nEnter date of booking :-\t");
  scanf("%d", &key);

  printf("\n Enter user id number :-\t");

  scanf("%d", &data);

  insert(key, data);

  break;

case 2:

  printf("\n\n Enter the date of booking to be deleted-:");

  scanf("%d", &key);

  remove_element(key);

  break;
```

```
    case 3:

      n = size_of_hashtable();
      printf("\n Number of books are-:%d\n", n);

      break;

    case 4:

      display();

      break;

    default: printf("Invalid Input\n");
          break;



    }

    printf("\n\n Do you want to continue (press 1 for yes): ");
    scanf("%d", &c);

  } while (c == 1);

  free(array);
}

void print_area()
{
   FILE *fp=fopen("cities.txt","r");

   if(fp==NULL);
   {
       printf("unable to read the file....");

   }

   int k=0;

   printf("\n DISPLAYING AREAS OF THE BELAGAVI CITY>>>>>>>>>>>>>>>>>>>");

    printf("\n -:The the areas of belagavi under consideration of the shortest path:-");


    printf("\n AREA ID | AREA NAME ");

    printf("\n-------------------------------------------------");


    while (fscanf(fp, "%d%s", &c[k].city_id, c[k].city_name) == 2)
    {
        printf("\n %d.%s", c[k].city_id, c[k].city_name);
        k++;
    }

    fclose(fp);



}
```

```
int user_program_advanced()
{

    int choice, src, dest,login;
    if (authenticate_user())
        {
            printf("\n 1.Choosing the events ");
            printf("\n 2.venue hall list ");
            printf("\n 3.travel to the venue ");
            printf("\n 4.book the venue ");
            printf("\n 5.Find Shortest Path");
            printf("\n 6.exit");

        while (1)
        {
            print_pattern();



            printf("\n\n Enter your choice: ");
            scanf("%d", &choice);

            switch (choice)
            {
            case 1:
                a = load_events();
                break;
            case 2:
                venue_halls();
                break;
            case 3:
```

```
          travel_venue();
          break;
        case 4: book_venue();


          break;
        case 5:load_adjacency();
          printf("\n\n Enter source city id: ");
          scanf("%d", &src);
          printf("\n\n Enter destination city id: ");
          scanf("%d", &dest);
          print_area();
          dijkstra(src, dest);
          break;
        case 6:
          exit(1);
          break;
        default:
          printf("\n\n Invalid choice. Please try again.\n");
          break;
        }
    }
}



  else


{


  printf("\n INVALID USER NAME OR PASSWORD... !! TRY AGAIN");
  return 1;
}
```

```
}
  // User interface program
  int user_program()
  {

    int choice, src, dest,login;

  printf("\n1.SIGNUP TO THE SITE");
  printf("\n2.LOGIN TO THE SITE");

  printf("\nChoose your option:-");

  scanf("%d",&login);


  switch(login)
  {

   case 1:  user_signup();

          break;


    case 2: user_program_advanced();

          break;


    default : printf("\n INVALID CHOICE");
```

```
            exit(0);

    }

    }
    // Add particular event:
    int add_event()
    {

        FILE *fp1;
        fp1 = fopen("event1.txt", "r+");

        if (fp1 == NULL)
        {
            printf("\n File is not found");

            return 1;
        }

        printf("\n DISPLAYING THE EVENTS LIST >>>>>>>>>>>>>>>>>>>>>>");

        printf("\n\n\n  -:The events at UK 27 are :-");

        print_pattern();

        printf("\n EVENT ID | EVENT NAME ");

        printf("\n -------------------------------------");

        print_pattern();
        while (fscanf(fp1, "%d %s", &e[i].Id, e[i].event_name) == 2)
        {
            printf("\n %d.%s", e[i].Id, e[i].event_name);
            i++;
        }
        int id;
        char event[50];

        printf("\n\n Enter the next events id:");

        scanf("%d",&id);

        printf("\n Enter the event name:");

        scanf("%s",event);

        fprintf(fp1,"\n %d %s",id,event);

        fclose(fp1);

        printf("\n Events has been successfully added to the list....... ");


    }



    int add_venue_hall()
    {
```

```
FILE *fp1 = fopen("venues.txt", "r+");

 if (fp1 == NULL)
 {
    printf("\n File is not found");
    return 1;
 }

 printf("\nDISPLAYING VENUE HALLS AT THE VENUE >>>>>>>>>>>>>>>>>>");

 printf("\n\n\n -:The different venue hall at UK 27 are :-");

 print_pattern();

 printf("\n VENUE ID | VENUE HALL NAME | PRICE ");

 printf("\n--------------------------------------------");


 while (fscanf(fp1, "\n%d %s %d", &v[j].venue_id, v[j].venue_hall, &v[j].price) == 3)
 {
     printf("\n %d.%s %d", v[j].venue_id, v[j].venue_hall, v[j].price);
    j++;
 }

 int id,price;
 char venue_h[50];
 printf("\n\n Enter the next events id:");

 scanf("%d",&id);
```

```
   printf("\n\n Enter the event name:");

   scanf("%s",venue_h);

   printf("\n\n Event hall price:");

   scanf("%d",&price);

   fprintf(fp1,"\n %d %s %d",id,venue_h,price);

   fclose(fp1);

    printf("\n venue hall has been successfully added to the list....... ");

}

int booking_data_display()
{
   FILE *fp=fopen("hash_table_data.txt","r");

     if (fp == NULL)
   {
      printf("\n File is not found");

      return 1;
   }

   print_line();

   printf("\n BOOKING DETAILS HERE  >>>>>>>>>>>>>> ");
```

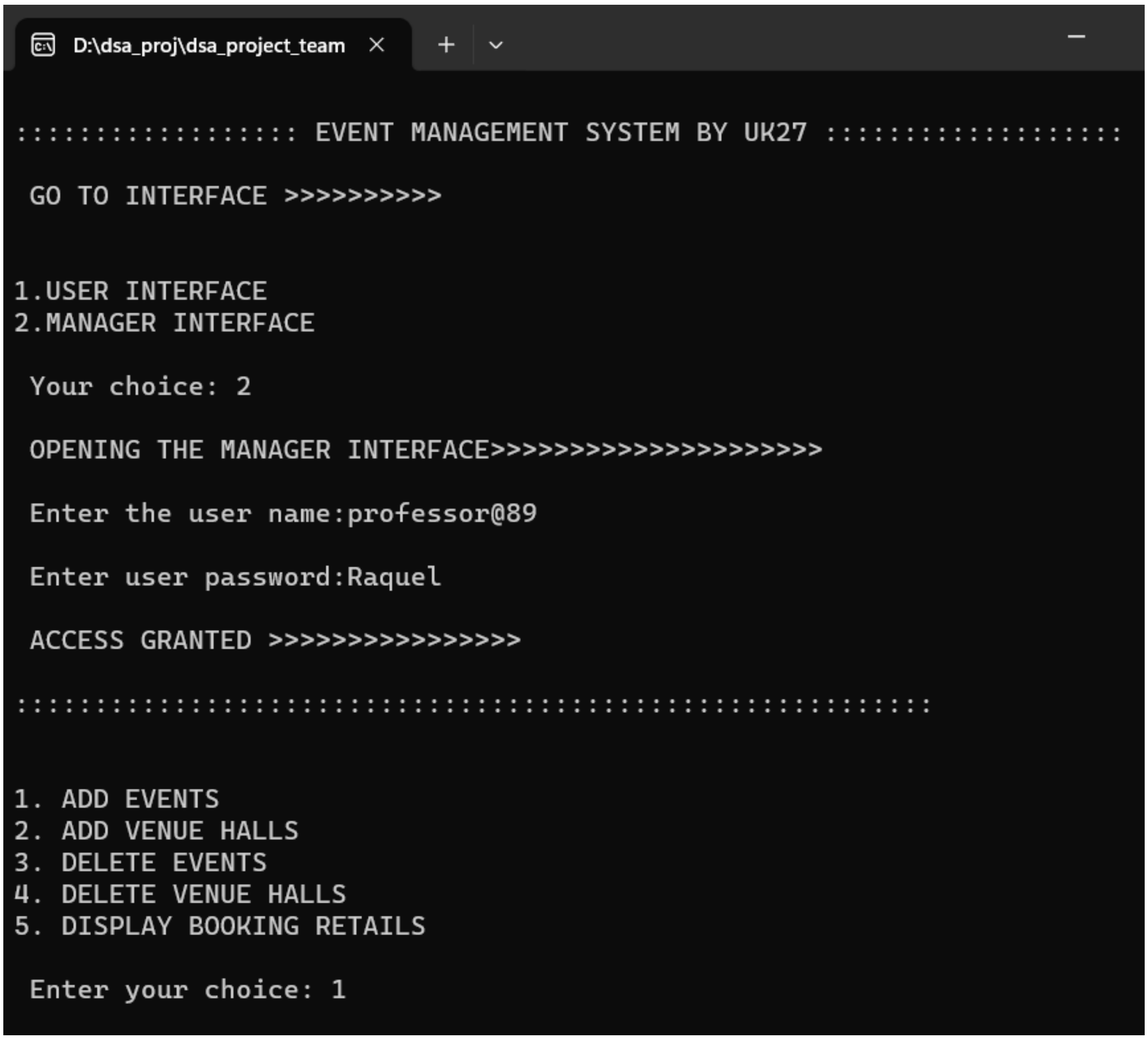
```
    printf("\n\n\n DATE\tUSER ID");


    printf("\n---------------------------");


    int i=0;


    while(fscanf(fp,"%d %d",&m[i].date,&m[i].user_id)==2)
    {
        printf("\n %d\t%d",m[i].date,m[i].user_id);
        i++;
    }


}


int manager_login_info()
{
    char user_name[50],password[50];


    printf("\n Enter the user name:");
    scanf("%s",user_name);


    printf("\n Enter user password:");
    scanf("%s",password);


    if(strcmp(user_name,"professor@89")==0 && strcmp(password,"Raquel")==0)
    {
        printf("\n ACCESS GRANTED >>>>>>>>>>>>>>>>");


        return 1;
    }
    else
```

```
        {
            return 0;
        }

    }



int display_events()
{
    FILE *fp1;
        fp1 = fopen("event1.txt", "r");

        if (fp1 == NULL)
        {
            printf("\n File is not found");
            return 1;
        }

        print_pattern();

        printf("\n DISPLAYING EVENTS LIST >>>>>>>>>>>>>>>>>>>> ");

        print_line();

        printf("\n -:The events at UK 27 are :-");

        print_pattern();

        printf("\n EVENT ID | EVENT NAME | PRICE");

        printf("\n ------------------------------------------");
        while (fscanf(fp1, "%d %s", &e[i].Id, e[i].event_name) == 2)

        {

            printf("\n %d.%s", e[i].Id, e[i].event_name);

            i++;

        }


        fclose(fp1);


}


int display_venuehalls()

{

    FILE *fp1 = fopen("venues.txt", "r");


        if (fp1 == NULL)

        {

            printf("\n File is not found");

            return 1;

        }


        print_pattern();


        printf("\n DISPLAYING VENUE HALLS LIST>>>>>>>>>>>>>>>>>>>>");



        printf("\n\n\n -: The different venue hall at UK 27 are :-");



        print_pattern();
        printf("\n VENUE ID | VENUE HALL NAME | PRICE");

        printf("\n --------------------------------------------------");



        while (fscanf(fp1, "%d %s %d", &v[j].venue_id, v[j].venue_hall, &v[j].price) == 3)
        {
            printf("\n %d.%s %d", v[j].venue_id, v[j].venue_hall, v[j].price);
            j++;
        }




        fclose(fp1);

}



    void manager_program() {
        int choice;



        if(manager_login_info())

      {
        printf("\n1.DISPLAY EVENTS LIST");


        printf("\n2. ADD EVENTS");


        printf("\n3.DISPLAY VENUE HALLS LIST");
printf("\n4. ADD VENUE HALLS");

printf("\n5. DELETE EVENTS");

printf("\n6. DELETE VENUE HALLS");

printf("\n7. DISPLAY BOOKING RETAILS");

while (1)

{  print_pattern();




  printf("\n\n Enter your choice: ");
   scanf("%d", &choice);

   switch (choice)

  {
       case 1:
           display_events();
               break;

       case 2:
           add_event();
           break;

       case 3: display_venuehalls();
              break;
  }



    case 4:

       add_venue_hall();

       break;


    case 5:

       printf("\nEnter the event ID to delete: ");

       scanf("%d", &a);

       delete_event(a);

       break;


    case 6:

       printf("\nEnter the venue ID to delete: ");

       scanf("%d", &b);

       delete_venue(b);

       break;


    case 7:

       booking_data_display();

       break;


    case 8:printf("\n PROCESS COMPLETED THANK TOU FOR VISITING ....!!");

         exit(0);




    default:

       printf("\nInvalid choice. Please try again.\n");

       break;

 }
```

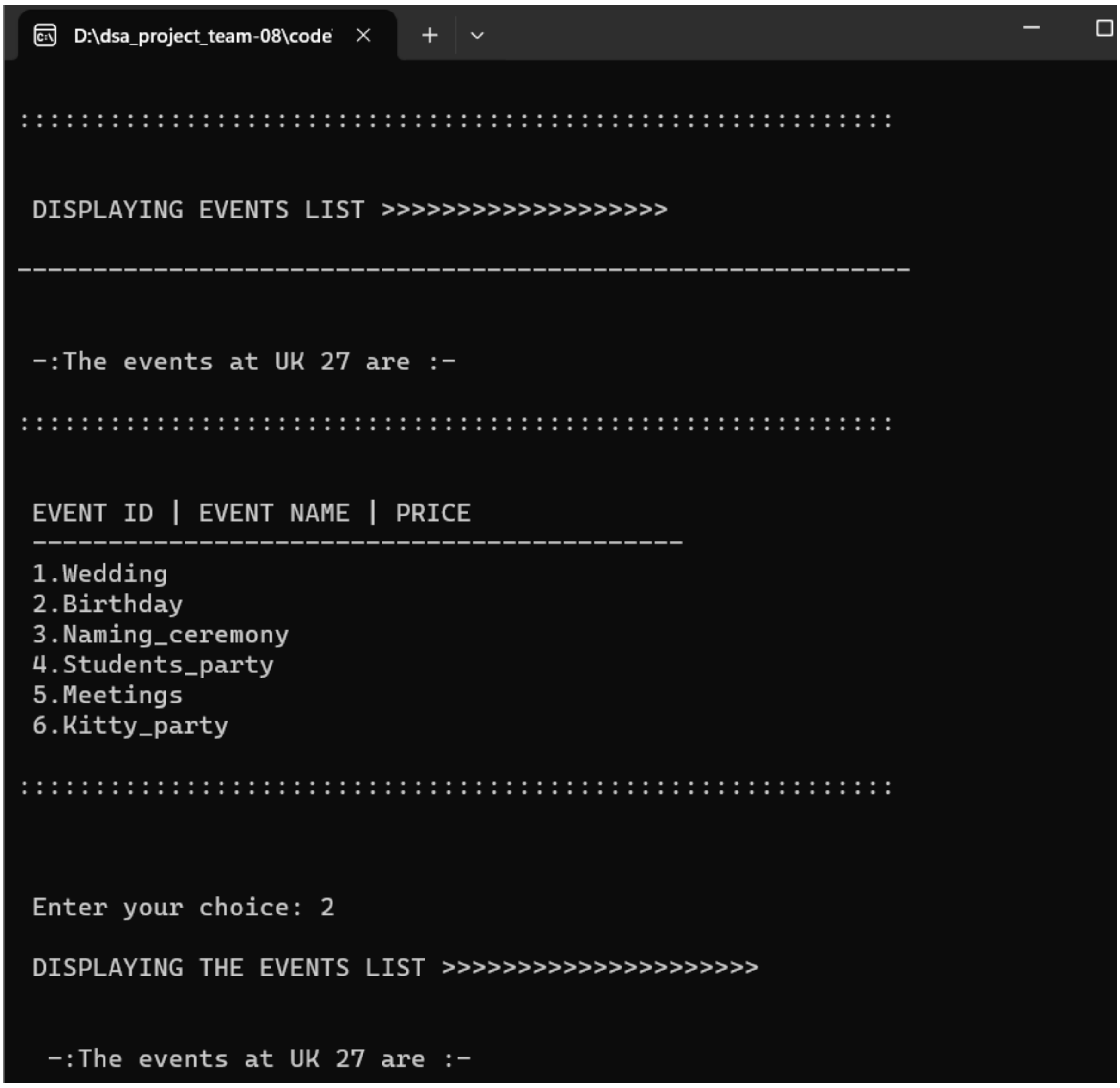
```
    }


  }
     else
    {
        printf("\n INVALID USER NAME OR PASSWORD TRY AGAIN");
    }
}




void content_display()
{
       printf("\n:::::::::::::::::: EVENT MANAGEMENT SYSTEM BY UK27 ::::::::::::::::::\n");
}



void user_choice()
{
    int choice_control;


    printf("\n GO TO INTERFACE >>>>>>>>>>\n\n ");




    printf("\n1.USER INTERFACE ");
    printf("\n2.MANAGER INTERFACE");




    printf("\n\n Your choice: ");
    scanf("%d",&choice_control);


    while(1)
    {
        switch (choice_control)
        {


        case 1:
              printf("\n OPENING THE USER INTERFACE>>>>>>>>>>>>>>>>>>>>>>>> \n ");
              user_program();
              break;


        case 2:
               printf("\n OPENING THE MANAGER INTERFACE>>>>>>>>>>>>>>>>>>>>>>\n");
               manager_program();


              break;




             default: printf("\n Varify your entry........!!");


                     break;
        }
    }
}




int main()
{
  content_display();
  user_choice();


   return 0;


}
```

PROGRAM OUTPUT

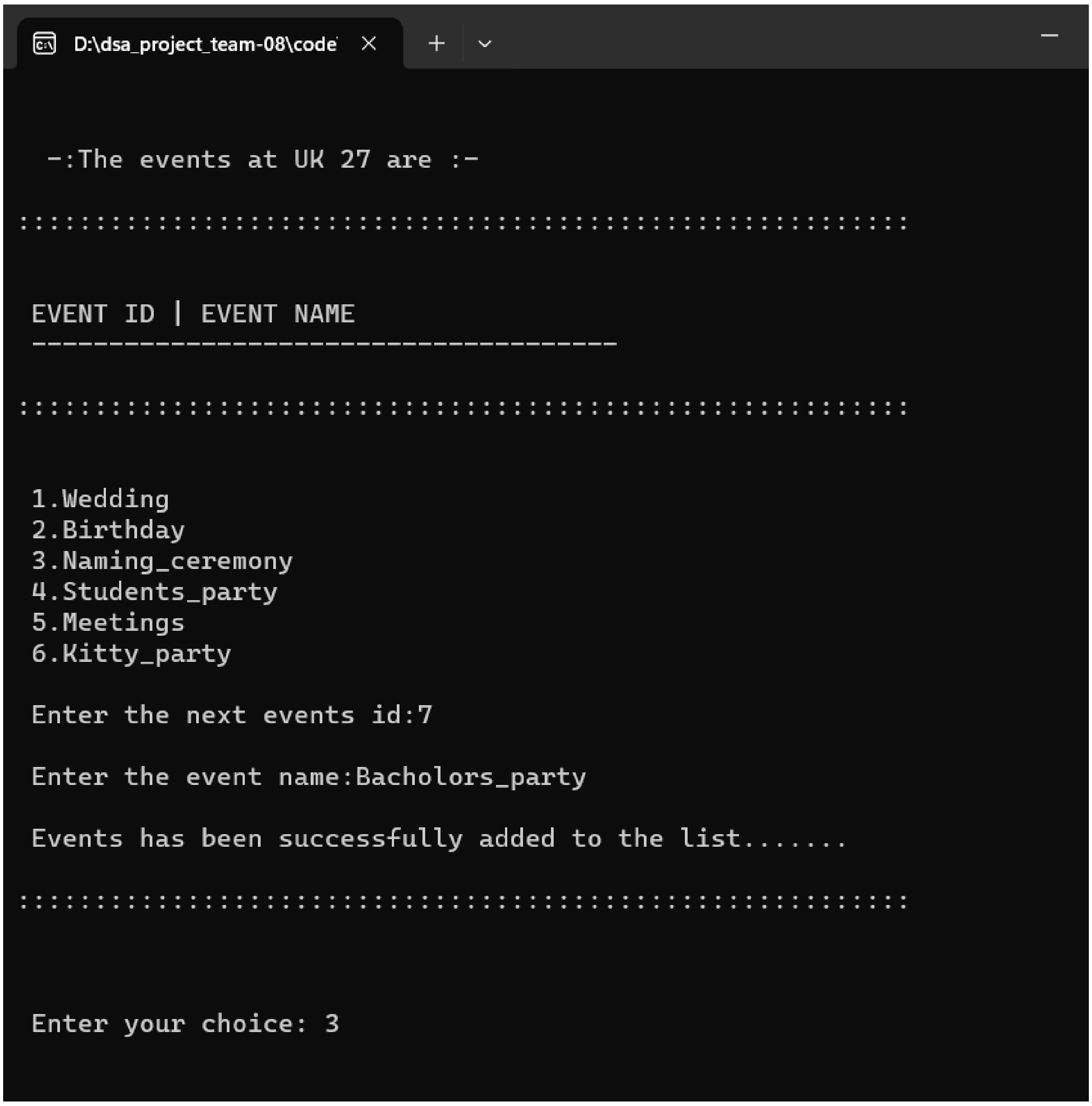
1) MANAGER INTERFACE



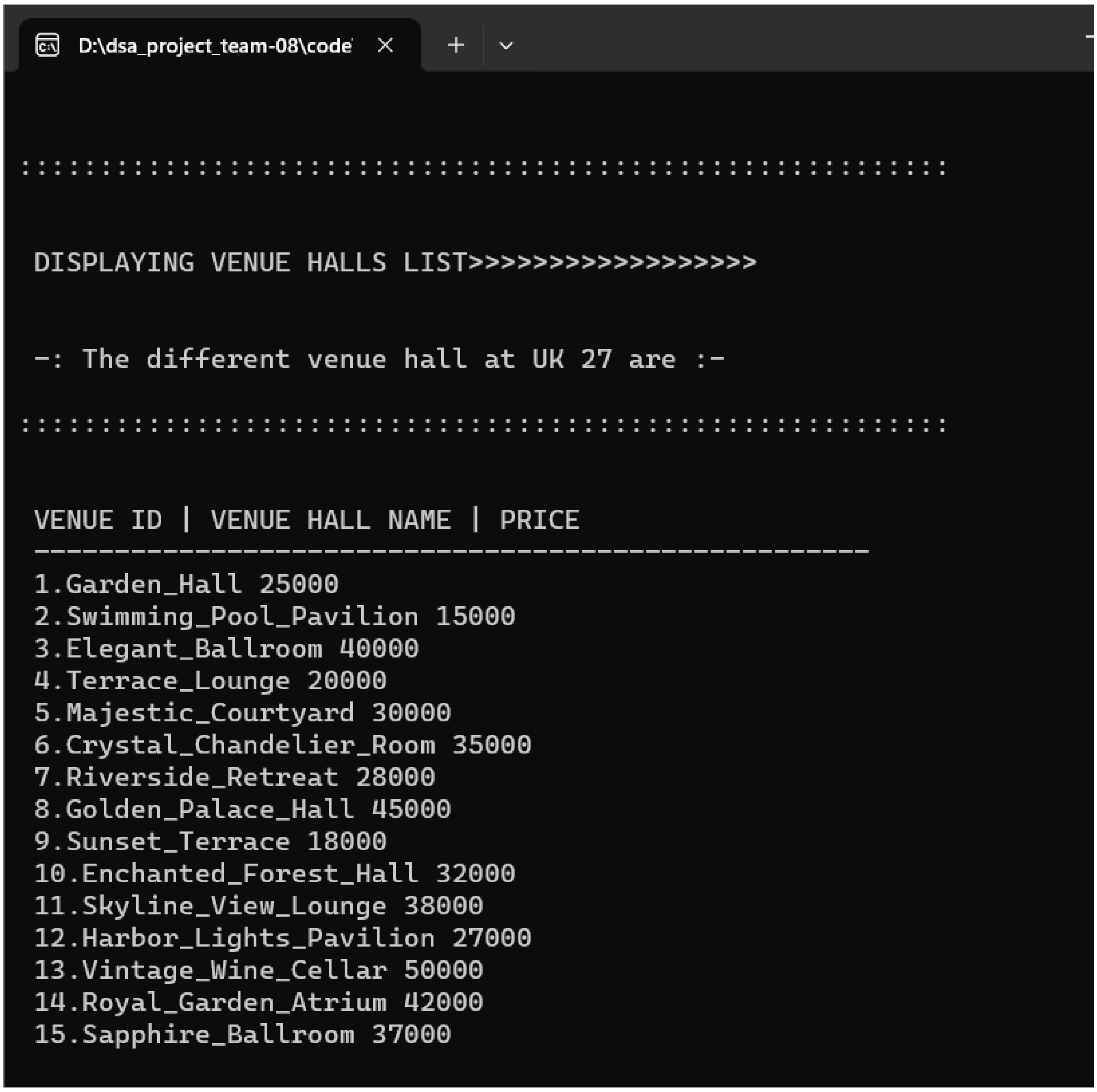
```
:::::::::::::::::: EVENT MANAGEMENT SYSTEM BY UK27 :::::::::::::::::::

 GO TO INTERFACE >>>>>>>>>>


1.USER INTERFACE
2.MANAGER INTERFACE

 Your choice: 2

 OPENING THE MANAGER INTERFACE>>>>>>>>>>>>>>>>>>>>>

 Enter the user name:professor@89

 Enter user password:Raquel

 ACCESS GRANTED >>>>>>>>>>>>>>>

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


1. ADD EVENTS
2. ADD VENUE HALLS
3. DELETE EVENTS
4. DELETE VENUE HALLS
5. DISPLAY BOOKING RETAILS

 Enter your choice: 1
```



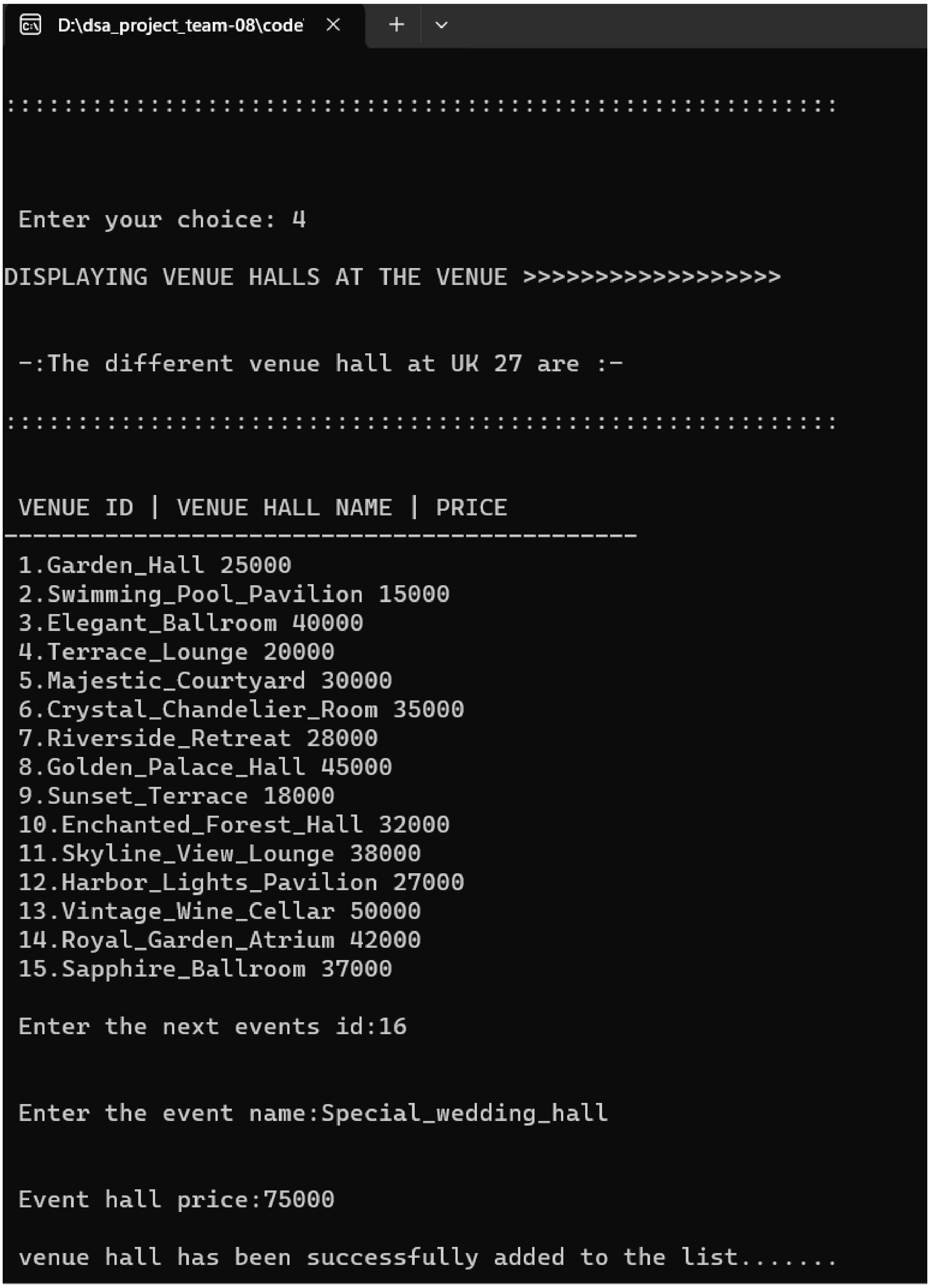
```
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 DISPLAYING EVENTS LIST >>>>>>>>>>>>>>>>>>>

-----------------------------------------------------------


 -:The events at UK 27 are :-

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 EVENT ID | EVENT NAME | PRICE
 -------------------------------------------
 1.Wedding
 2.Birthday
 3.Naming_ceremony
 4.Students_party
 5.Meetings
 6.Kitty_party

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 2

 DISPLAYING THE EVENTS LIST >>>>>>>>>>>>>>>>>>>>>


  -:The events at UK 27 are :-
```



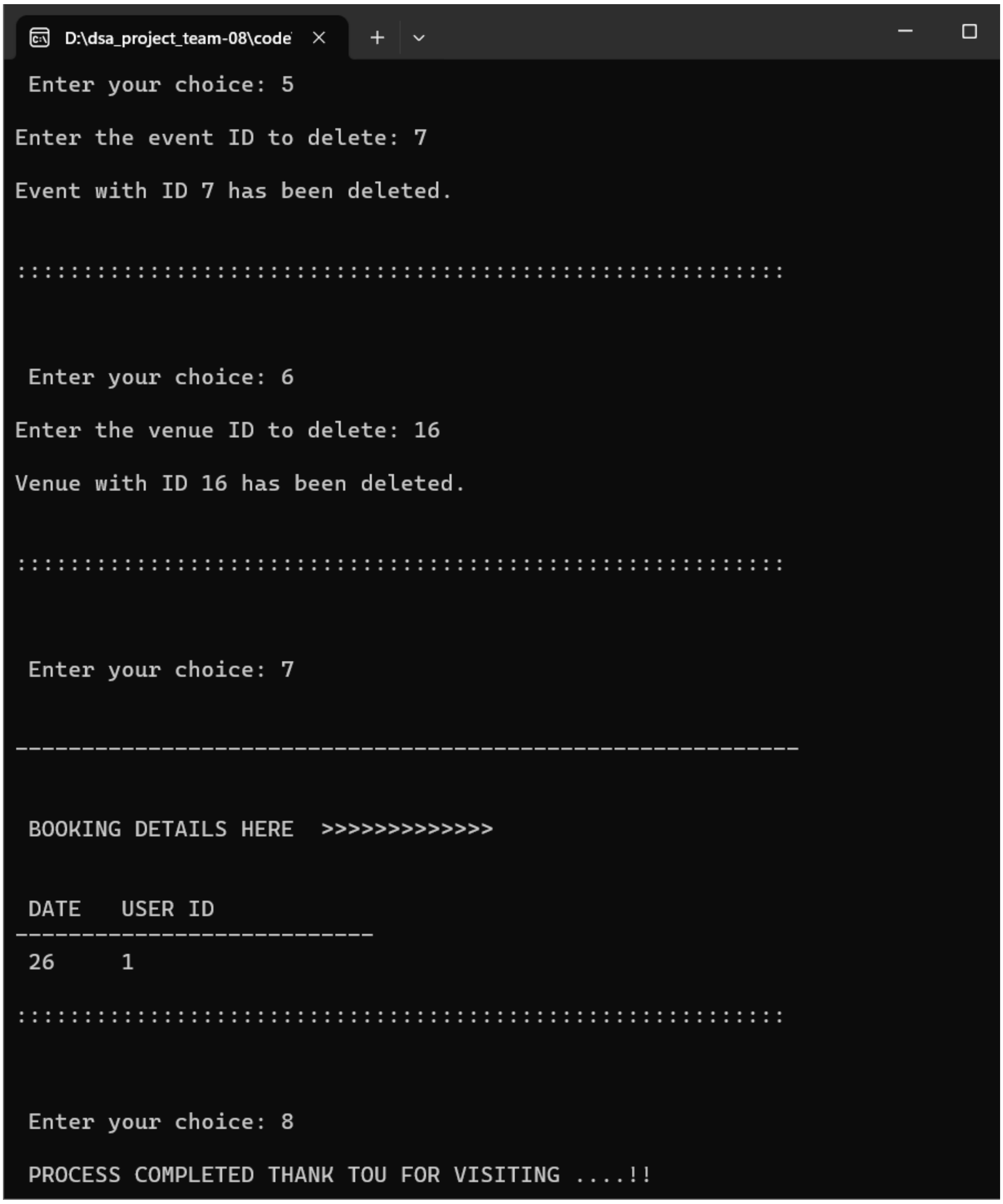
```
  -:The events at UK 27 are :-

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 EVENT ID | EVENT NAME
 --------------------------------------

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 1.Wedding
 2.Birthday
 3.Naming_ceremony
 4.Students_party
 5.Meetings
 6.Kitty_party

 Enter the next events id:7

 Enter the event name:Bacholors_party

 Events has been successfully added to the list.......

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 3
```



```
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 DISPLAYING VENUE HALLS LIST>>>>>>>>>>>>>>>>>>>


 -: The different venue hall at UK 27 are :-

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 VENUE ID | VENUE HALL NAME | PRICE
 ----------------------------------------------------
 1.Garden_Hall 25000
 2.Swimming_Pool_Pavilion 15000
 3.Elegant_Ballroom 40000
 4.Terrace_Lounge 20000
 5.Majestic_Courtyard 30000
 6.Crystal_Chandelier_Room 35000
 7.Riverside_Retreat 28000
 8.Golden_Palace_Hall 45000
 9.Sunset_Terrace 18000
 10.Enchanted_Forest_Hall 32000
 11.Skyline_View_Lounge 38000
 12.Harbor_Lights_Pavilion 27000
 13.Vintage_Wine_Cellar 50000
 14.Royal_Garden_Atrium 42000
 15.Sapphire_Ballroom 37000
```

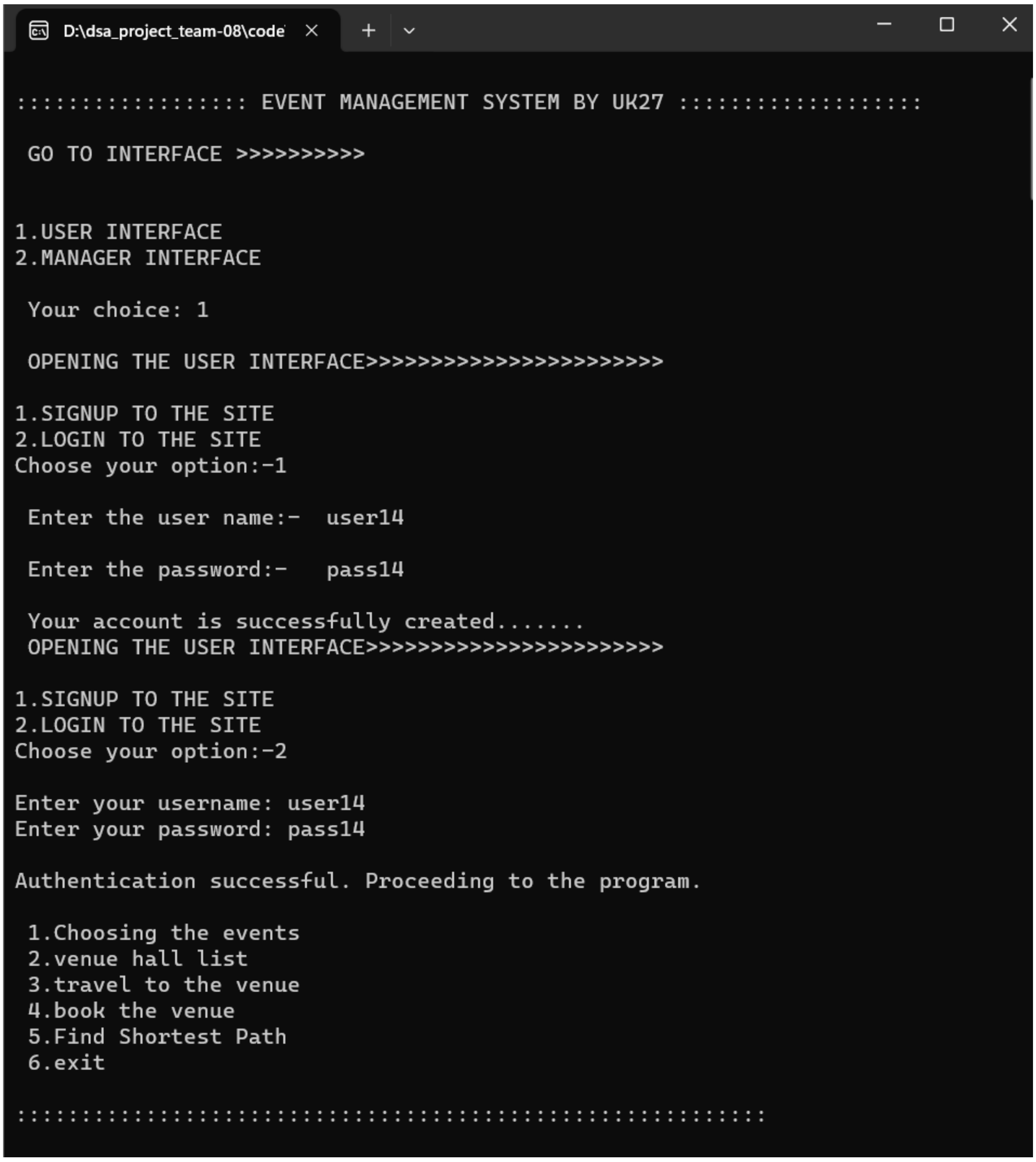


```
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 4

DISPLAYING VENUE HALLS AT THE VENUE >>>>>>>>>>>>>>>>>>>>


 -:The different venue hall at UK 27 are :-

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


 VENUE ID | VENUE HALL NAME | PRICE
--------------------------------------------
 1.Garden_Hall 25000
 2.Swimming_Pool_Pavilion 15000
 3.Elegant_Ballroom 40000
 4.Terrace_Lounge 20000
 5.Majestic_Courtyard 30000
 6.Crystal_Chandelier_Room 35000
 7.Riverside_Retreat 28000
 8.Golden_Palace_Hall 45000
 9.Sunset_Terrace 18000
 10.Enchanted_Forest_Hall 32000
 11.Skyline_View_Lounge 38000
 12.Harbor_Lights_Pavilion 27000
 13.Vintage_Wine_Cellar 50000
 14.Royal_Garden_Atrium 42000
 15.Sapphire_Ballroom 37000

 Enter the next events id:16


 Enter the event name:Special_wedding_hall


 Event hall price:75000

 venue hall has been successfully added to the list.......
```

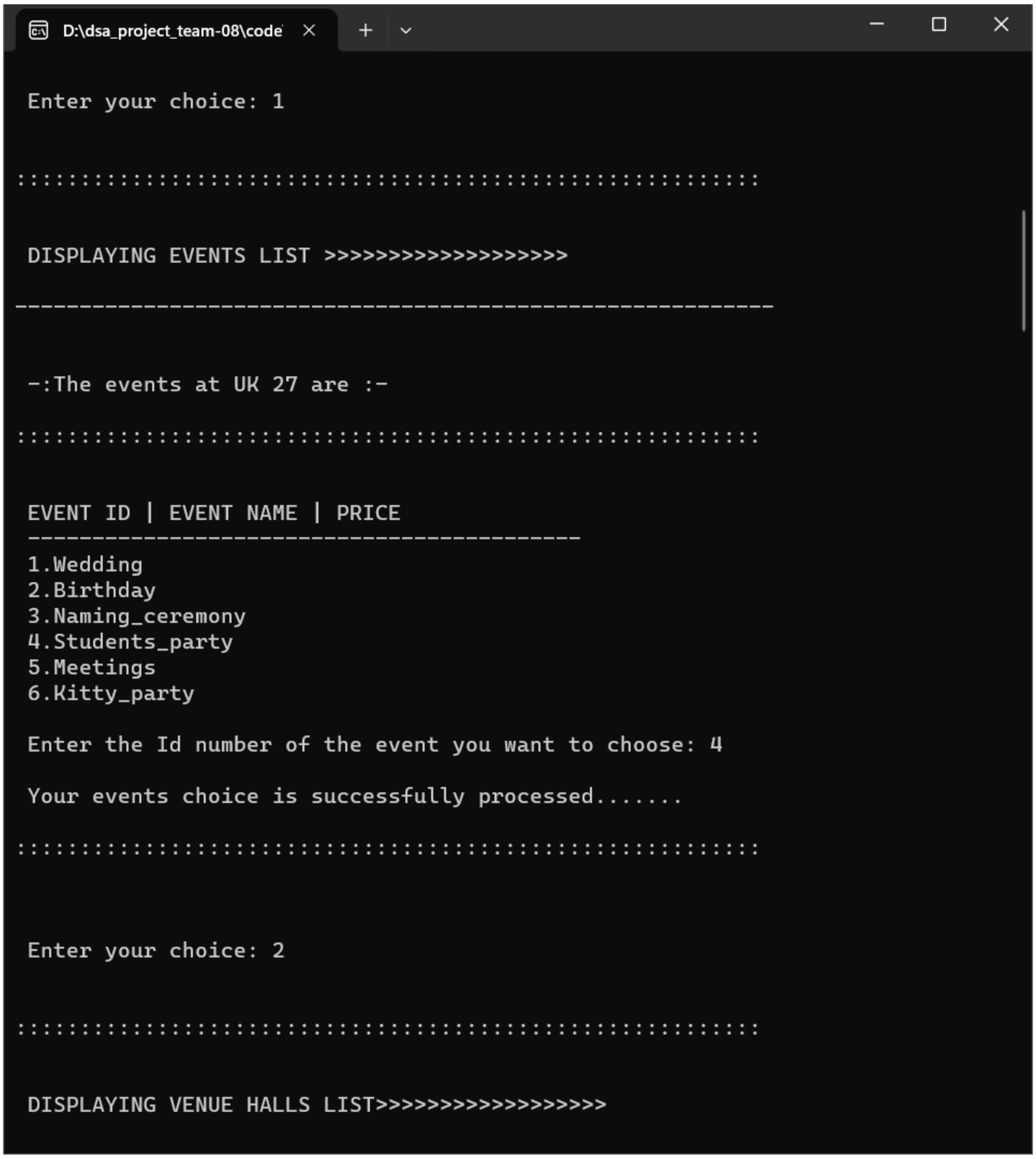


```
 Enter your choice: 5

Enter the event ID to delete: 7

Event with ID 7 has been deleted.


::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 6

Enter the venue ID to delete: 16

Venue with ID 16 has been deleted.


::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 7


-----------------------------------------------------------


 BOOKING DETAILS HERE  >>>>>>>>>>>>>


 DATE   USER ID
---------------------------
 26     1

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 8

 PROCESS COMPLETED THANK TOU FOR VISITING ....!!
```

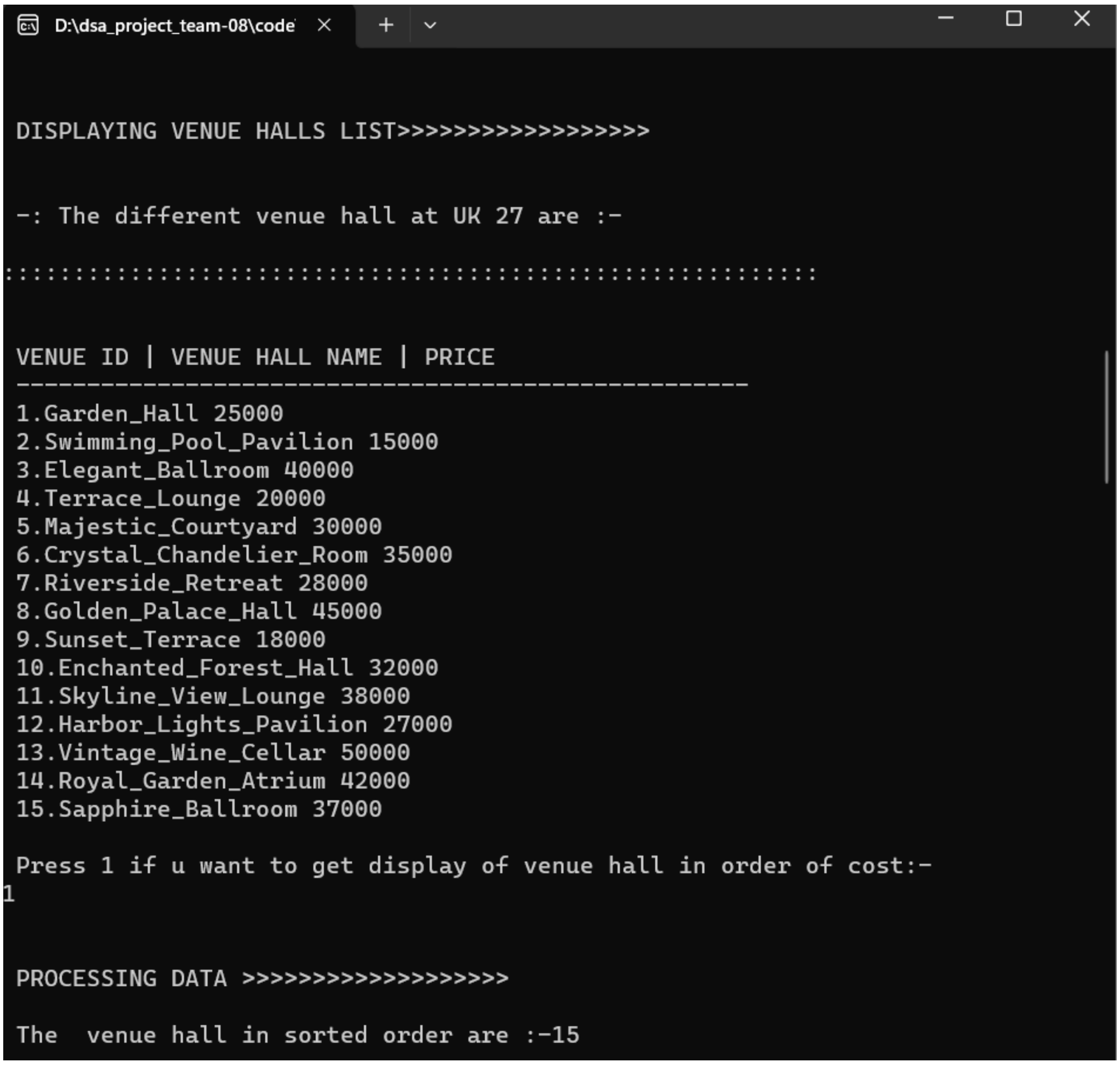
2)USER INTERFACE



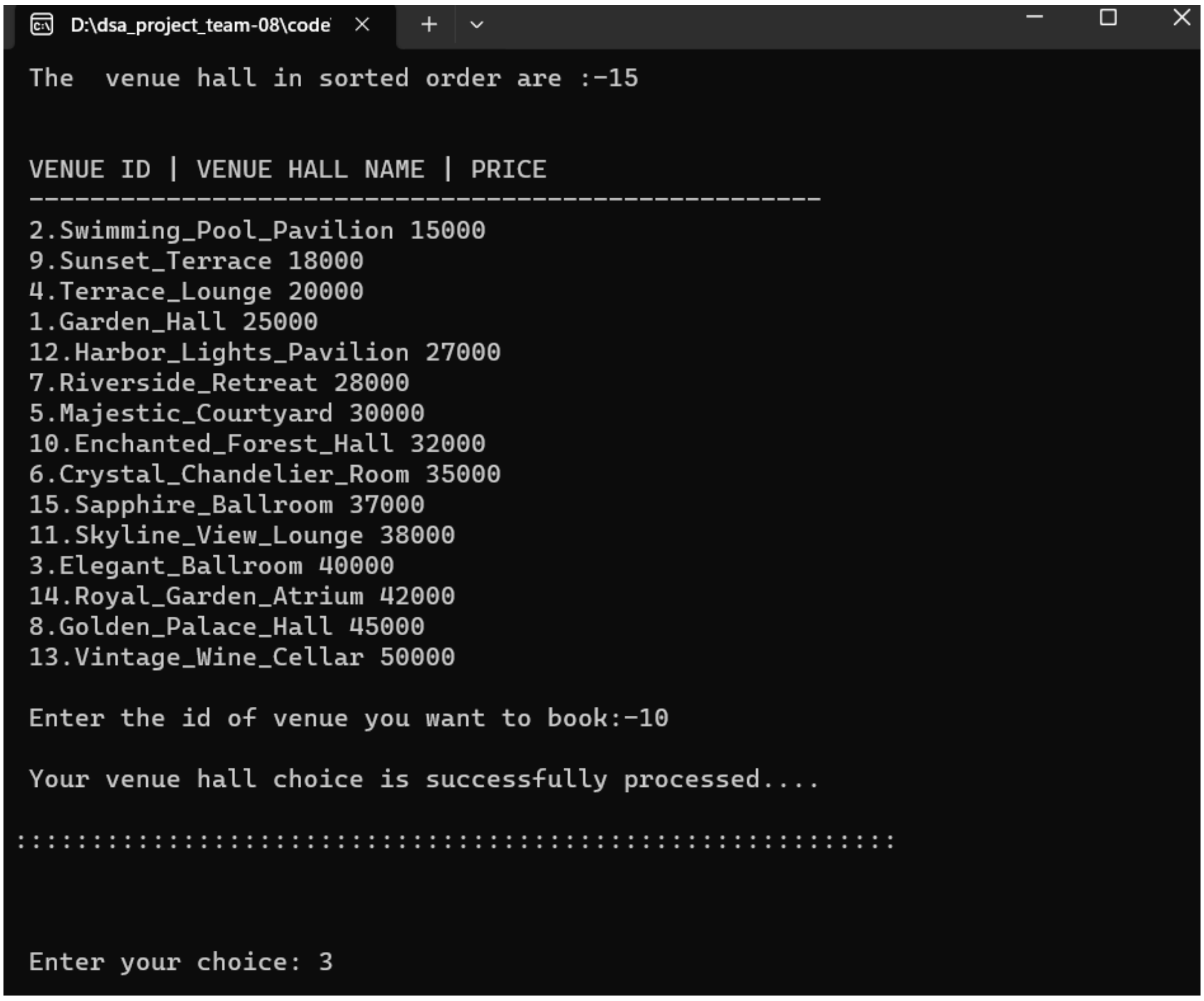
```
:::::::::::::::::: EVENT MANAGEMENT SYSTEM BY UK27 ::::::::::::::::::

 GO TO INTERFACE >>>>>>>>>>


1.USER INTERFACE
2.MANAGER INTERFACE

 Your choice: 1

 OPENING THE USER INTERFACE>>>>>>>>>>>>>>>>>>>>>>>

1.SIGNUP TO THE SITE
2.LOGIN TO THE SITE
Choose your option:-1

 Enter the user name:-  user14

 Enter the password:-   pass14

 Your account is successfully created.......
 OPENING THE USER INTERFACE>>>>>>>>>>>>>>>>>>>>>>>

1.SIGNUP TO THE SITE
2.LOGIN TO THE SITE
Choose your option:-2

Enter your username: user14
Enter your password: pass14

Authentication successful. Proceeding to the program.

 1.Choosing the events
 2.venue hall list
 3.travel to the venue
 4.book the venue
 5.Find Shortest Path
 6.exit

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
```



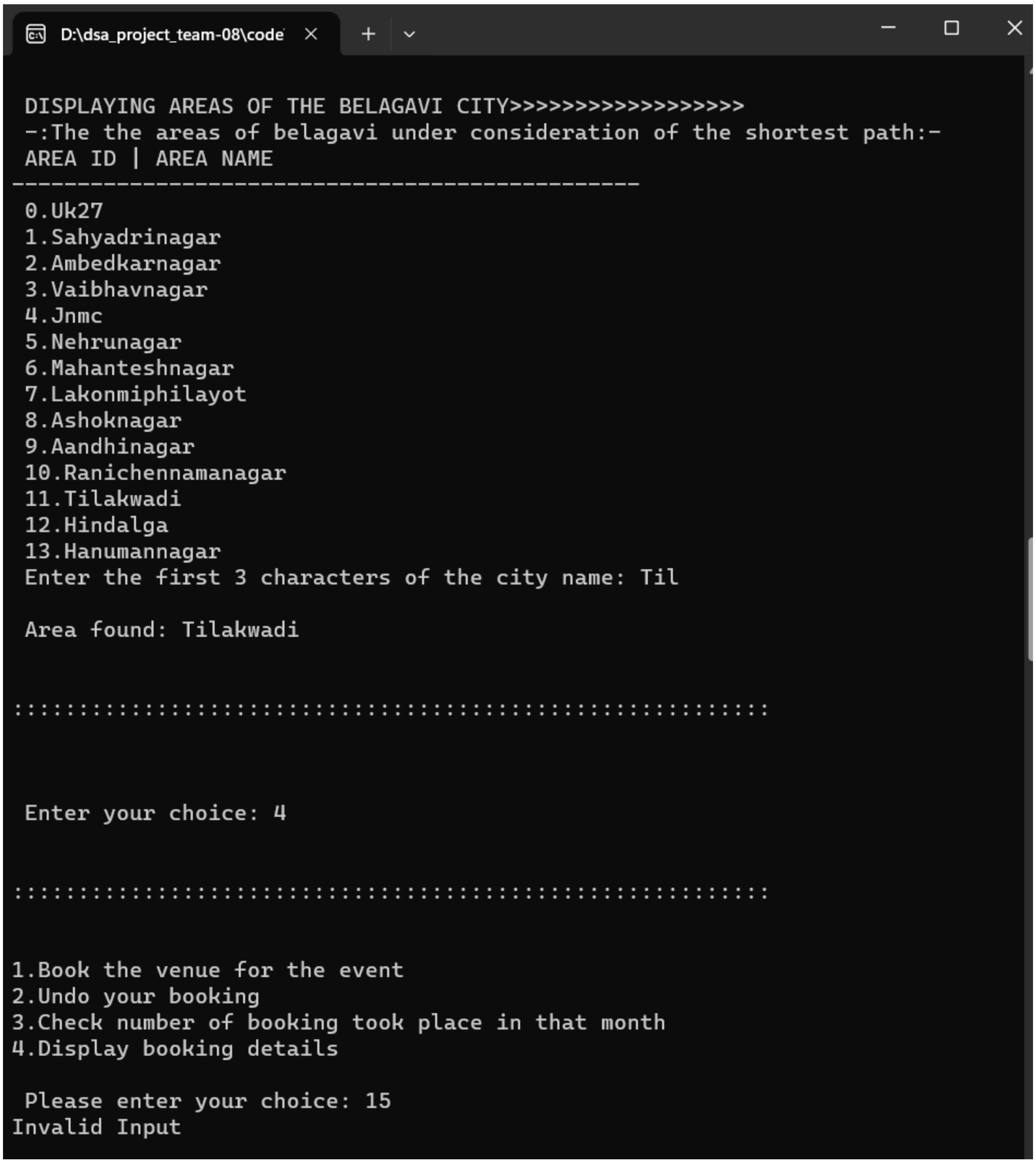
```
Enter your choice: 1


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


DISPLAYING EVENTS LIST >>>>>>>>>>>>>>>>>>>

-----------------------------------------------------------


-:The events at UK 27 are :-

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


EVENT ID | EVENT NAME | PRICE
-------------------------------------------
1.Wedding
2.Birthday
3.Naming_ceremony
4.Students_party
5.Meetings
6.Kitty_party

Enter the Id number of the event you want to choose: 4

Your events choice is successfully processed.......

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



Enter your choice: 2


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


DISPLAYING VENUE HALLS LIST>>>>>>>>>>>>>>>>>>
```



```
DISPLAYING VENUE HALLS LIST>>>>>>>>>>>>>>>>>>


-: The different venue hall at UK 27 are :-

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::


VENUE ID | VENUE HALL NAME | PRICE
----------------------------------------------------
1.Garden_Hall 25000
2.Swimming_Pool_Pavilion 15000
3.Elegant_Ballroom 40000
4.Terrace_Lounge 20000
5.Majestic_Courtyard 30000
6.Crystal_Chandelier_Room 35000
7.Riverside_Retreat 28000
8.Golden_Palace_Hall 45000
9.Sunset_Terrace 18000
10.Enchanted_Forest_Hall 32000
11.Skyline_View_Lounge 38000
12.Harbor_Lights_Pavilion 27000
13.Vintage_Wine_Cellar 50000
14.Royal_Garden_Atrium 42000
15.Sapphire_Ballroom 37000

Press 1 if u want to get display of venue hall in order of cost:-
1


PROCESSING DATA >>>>>>>>>>>>>>>>>>>

The  venue hall in sorted order are :-15
```

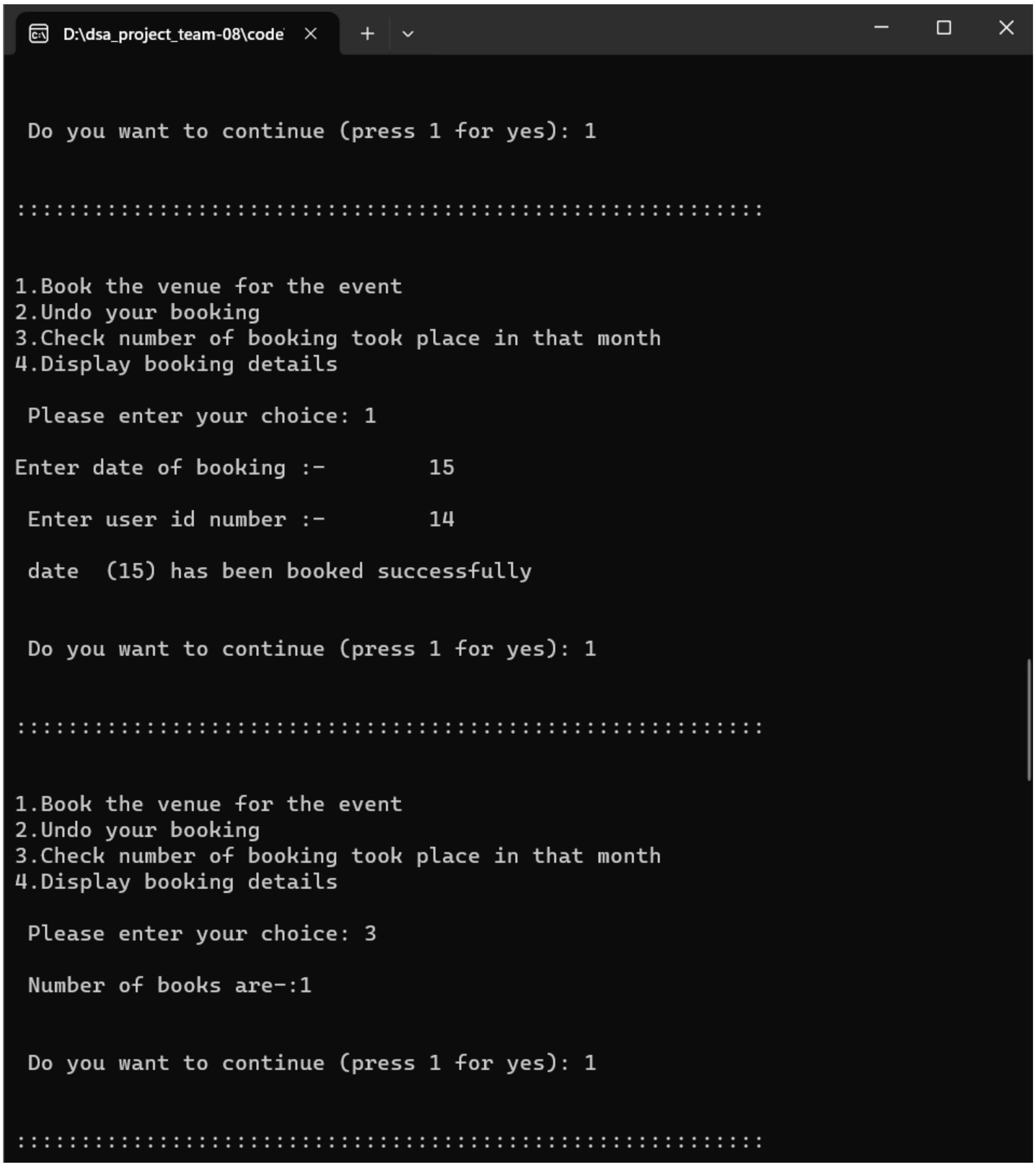


```
The  venue hall in sorted order are :-15


VENUE ID | VENUE HALL NAME | PRICE
----------------------------------------------------
2.Swimming_Pool_Pavilion 15000
9.Sunset_Terrace 18000
4.Terrace_Lounge 20000
1.Garden_Hall 25000
12.Harbor_Lights_Pavilion 27000
7.Riverside_Retreat 28000
5.Majestic_Courtyard 30000
10.Enchanted_Forest_Hall 32000
6.Crystal_Chandelier_Room 35000
15.Sapphire_Ballroom 37000
11.Skyline_View_Lounge 38000
3.Elegant_Ballroom 40000
14.Royal_Garden_Atrium 42000
8.Golden_Palace_Hall 45000
13.Vintage_Wine_Cellar 50000

Enter the id of venue you want to book:-10

Your venue hall choice is successfully processed....

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



Enter your choice: 3
```

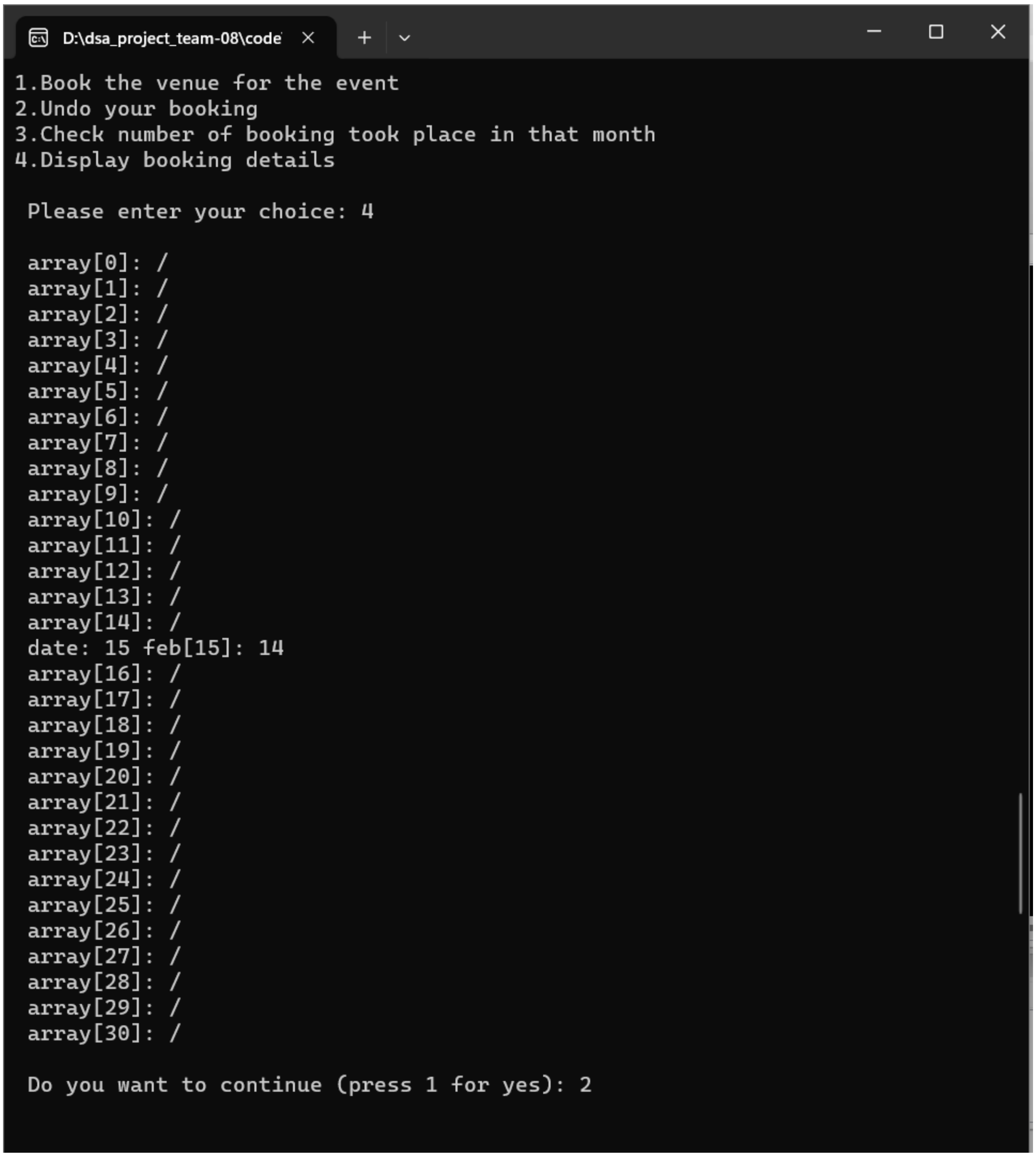


D:\dsa_project_team-08\code

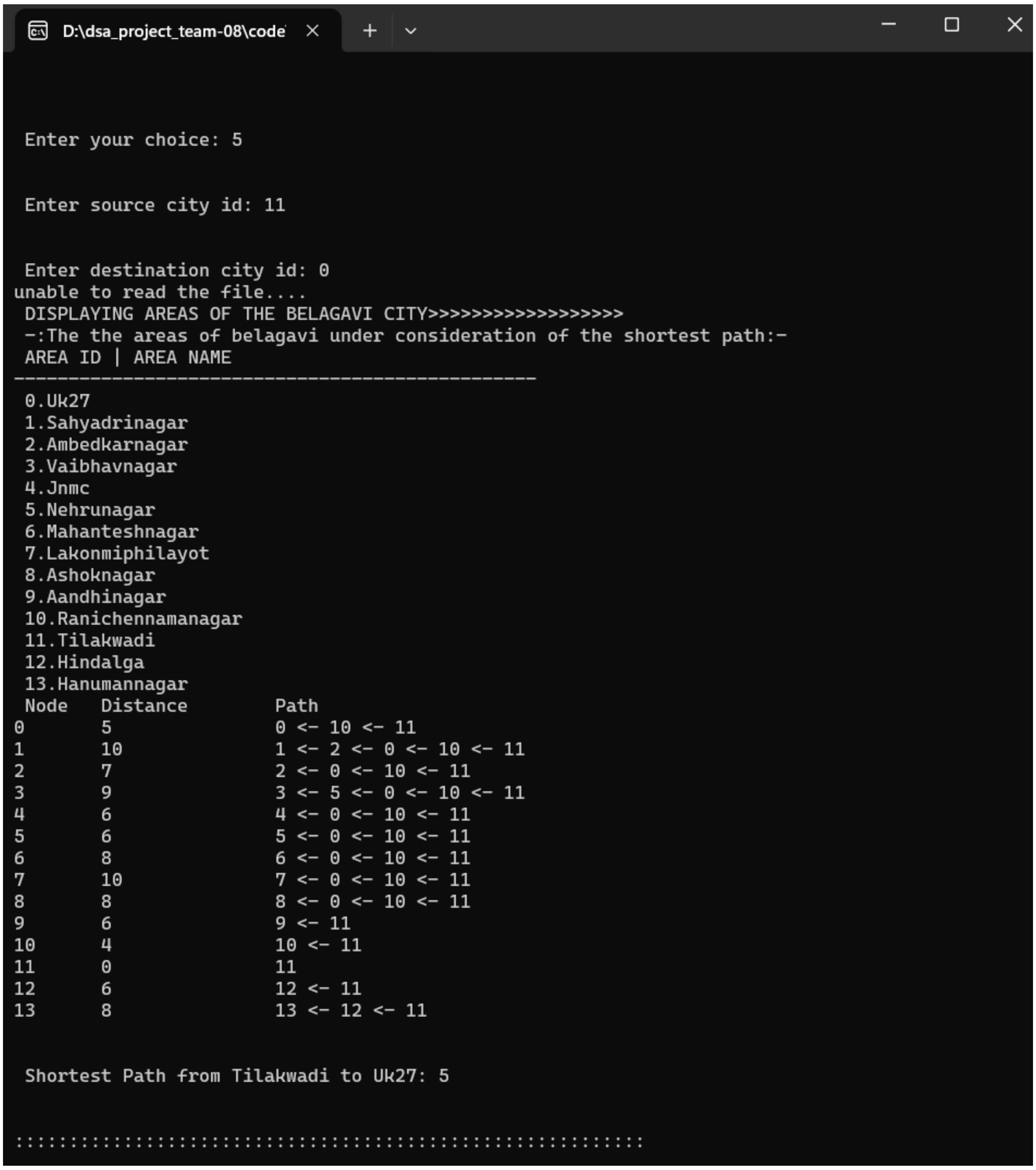
```
 DISPLAYING AREAS OF THE BELAGAVI CITY>>>>>>>>>>>>>>>>>>
 -:The the areas of belagavi under consideration of the shortest path:-
 AREA ID | AREA NAME
------------------------------------------------
 0.Uk27
 1.Sahyadrinagar
 2.Ambedkarnagar
 3.Vaibhavnagar
 4.Jnmc
 5.Nehrunagar
 6.Mahanteshnagar
 7.Lakonmiphilayot
 8.Ashoknagar
 9.Aandhinagar
 10.Ranichennamanagar
 11.Tilakwadi
 12.Hindalga
 13.Hanumannagar
 Enter the first 3 characters of the city name: Til

 Area found: Tilakwadi


::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 4


::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


1.Book the venue for the event
2.Undo your booking
3.Check number of booking took place in that month
4.Display booking details

 Please enter your choice: 15
Invalid Input
```



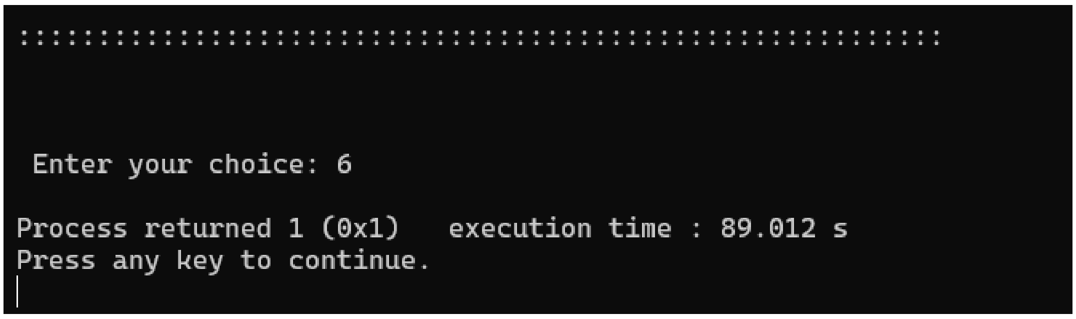
```
 Do you want to continue (press 1 for yes): 1


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


1.Book the venue for the event
2.Undo your booking
3.Check number of booking took place in that month
4.Display booking details

 Please enter your choice: 1

Enter date of booking :-        15

 Enter user id number :-        14

 date  (15) has been booked successfully


 Do you want to continue (press 1 for yes): 1


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::


1.Book the venue for the event
2.Undo your booking
3.Check number of booking took place in that month
4.Display booking details

 Please enter your choice: 3

 Number of books are-:1


 Do you want to continue (press 1 for yes): 1


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
```



```
1.Book the venue for the event
2.Undo your booking
3.Check number of booking took place in that month
4.Display booking details

 Please enter your choice: 4

 array[0]: /
 array[1]: /
 array[2]: /
 array[3]: /
 array[4]: /
 array[5]: /
 array[6]: /
 array[7]: /
 array[8]: /
 array[9]: /
 array[10]: /
 array[11]: /
 array[12]: /
 array[13]: /
 array[14]: /
 date: 15 feb[15]: 14
 array[16]: /
 array[17]: /
 array[18]: /
 array[19]: /
 array[20]: /
 array[21]: /
 array[22]: /
 array[23]: /
 array[24]: /
 array[25]: /
 array[26]: /
 array[27]: /
 array[28]: /
 array[29]: /
 array[30]: /

 Do you want to continue (press 1 for yes): 2
```



```
D:\dsa_project_team-08\code


 Enter your choice: 5


 Enter source city id: 11


 Enter destination city id: 0
unable to read the file....
 DISPLAYING AREAS OF THE BELAGAVI CITY>>>>>>>>>>>>>>>>>>
 -:The the areas of belagavi under consideration of the shortest path:-
 AREA ID | AREA NAME
------------------------------------------------
 0.Uk27
 1.Sahyadrinagar
 2.Ambedkarnagar
 3.Vaibhavnagar
 4.Jnmc
 5.Nehrunagar
 6.Mahanteshnagar
 7.Lakonmiphilayot
 8.Ashoknagar
 9.Aandhinagar
 10.Ranichennamanagar
 11.Tilakwadi
 12.Hindalga
 13.Hanumannagar
 Node   Distance        Path
0       5               0 <- 10 <- 11
1       10              1 <- 2 <- 0 <- 10 <- 11
2       7               2 <- 0 <- 10 <- 11
3       9               3 <- 5 <- 0 <- 10 <- 11
4       6               4 <- 0 <- 10 <- 11
5       6               5 <- 0 <- 10 <- 11
6       8               6 <- 0 <- 10 <- 11
7       10              7 <- 0 <- 10 <- 11
8       8               8 <- 0 <- 10 <- 11
9       6               9 <- 11
10      4               10 <- 11
11      0               11
12      6               12 <- 11
13      8               13 <- 12 <- 11


 Shortest Path from Tilakwadi to Uk27: 5


:::::::::::::::::::::::::::::::::::::::::::::::::::::::::
```



```
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::



 Enter your choice: 6

Process returned 1 (0x1)   execution time : 89.012 s
Press any key to continue.
```