A PROJECT ON

BUS RESERVATION SYSTEM

SUBMITTED BY:

Rinki Goyal
14CS038

SUBMITTED TO:

Mr.Sanjog Arora

ANAND INTERNATIONAL COLLEGE OF ENGG.
Department of COMPUTER SCIENCE

ANAND INTERNATIONAL COLLEGE OF ENGINEERING

C - PROJECT

CERTIFICATE

THIS IS TO CERTIFY THAT **RINKI GOYAL** *of* CS Branch, **ANAND INTERNATIONAL COLLEGE OF ENGG.** HAS SUCCESSFULLY COMPLETED THE PROJECT " **STUDENT ENQUIRY SYSTEM** " UNDER **Mr. SANJOG ARORA's** GUIDENCE .

Mr. Sanjog Arora
Assistant Professor

Mr. Dharmesh Santani
HOD (CSE)

ACKNOWLEDGEMENT

THIS PROJECT COULD NOT HAVE BEEN SUCCESSFULLY COMPLETED WITHOUT THE UNENDING AND CONTINIOUS SUPPORT AND HELP FROM ALL MY TEACHERS, ESPECIALLY **Mr. Sanjog Arora** . I WOULD ALSO LIKE TO THANK MY PARENTS FOR THEIR LOVE AND ENCOURAGEMENT THROUGHOUT THE MAKING OF THIS PROJECT.

INDEX

Week – 1

Tasks to be performed this week:

To study the following topics:

- File Handling
- Control Statements
- Functions
- Structures
- Pointers

Tasks performed this week:

- FILE HANDLING:

Through file handling, one can perform operations like create, modify, delete etc on system files.

File handling functions

Following functions that are popularly used in file handling :

a. fopen()

```
FILE *fopen(const char *path, const char *mode);
```

The fopen() function is used to open a file and associates an I/O stream with it. This function takes two arguments. The first argument is a pointer to a string containing name of the file to be opened while the second argument is the mode in which the file is to be opened. The mode can be :

- 'r' : Open text file for reading. The stream is positioned at the beginning of the file.
- 'r+' : Open for reading and writing. The stream is positioned at the beginning of the file.
- 'w' : Truncate file to zero length or create text file for writing. The stream is positioned at the beginning of the file.
- 'w+' : Open for reading and writing. The file is created if it does not exist, otherwise it is truncated. The stream is positioned at the beginning of the file.

- ‘a’ : Open for appending (writing at end of file). The file is created if it does not exist. The stream is positioned at the end of the file.
- ‘a+’ : Open for reading and appending (writing at end of file). The file is created if it does not exist. The initial file position for reading is at the beginning of the file, but output is always appended to the end of the file.

The fopen() function returns a FILE stream pointer on success while it returns NULL in case of a failure.

b. fread() and fwrite()

```
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);



size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream);
```

The functions fread/fwrite are used for reading/writing data from/to the file opened by fopen function. These functions accept three arguments. The first argument is a pointer to buffer used for reading/writing the data. The data read/written is in the form of ‘nmemb’ elements each ‘size’ bytes long.

In case of success, fread/fwrite return the number of bytes actually read/written from/to the stream opened by fopen function. In case of failure, a lesser number of byes (then requested to read/write) is returned.

c. fseek()

```
int fseek(FILE *stream, long offset, int whence);
```

The fseek() function is used to set the file position indicator for the stream to a new position. This function accepts three arguments. The first argument is the FILE stream pointer returned by the fopen() function. The second argument ‘offset’ tells the amount of bytes to seek. The third argument ‘whence’ tells from where the seek of ‘offset’ number of bytes is to be done. The available values for whence are SEEK_SET, SEEK_CUR, or SEEK_END. These three values (in order) depict the start of the file, the current position and the end of the file.

Upon success, this function returns 0, otherwise it returns -1.

d. fclose()

```
int fclose(FILE *fp);
```

The fclose() function first flushes the stream opened by fopen() and then closes the underlying descriptor. Upon successful completion this function returns 0 else end of file (eof) is returned. In case of failure, if the stream is accessed further then the behavior remains undefined.

• CONTROL STATEMENTS:

if statement

This is the most simple form of the branching statements.

It takes an expression in parenthesis and an statement or block of statements. if the expression is true then the statement or block of statements gets executed otherwise these statements are skipped.

NOTE: Expression will be assumed to be true if its evaulated values is non-zero.

if statements take the following form:

```
if (expression)
  statement;

or

if (expression)
  {
    Block of statements;
  }

or

if (expression)
  {
    Block of statements;
  }
```

```
else
  {
    Block of statements;
  }

or

if (expression)
  {
    Block of statements;
  }
else if(expression)
  {
    Block of statements;
  }
else
  {
    Block of statements;
  }
```

while loop

The most basic loop in C is the while loop.A while statement is like a repeating if statement. Like an If statement, if the test condition is true: the statments get executed. The difference is that after the statements have been executed, the test condition is checked again. If it is still true the statements get executed again.This cycle repeats until the test condition evaluates to false.

Basic syntax of while loop is as follows:

```
while ( expression )
{
   Single statement
   or
   Block of statements;
}
```

for loop

for loop is similar to while, it's just written differently. for statements are often used to proccess lists such a range of numbers:

Basic syntax of for loop is as follows:

```
for( expression1; expression2; expression3)
{
   Single statement
   or
   Block of statements;
}
```

In the above syntax:

- expression1 - Initialisese variables.

- expression2 - Condtional expression, as long as this condition is true, loop will keep executing.
- expression3 - expression3 is the modifier which may be simple increment of a variable.

switch case :

- Switch case statements are used to execute only specific case statements based on the switch expression.
- Below is the syntax for switch case statement.

```
switch (expression)
{
case label1:   statements;
break;
case label2:   statements;
break;
default:    statements;
break;
}
```

- <u>FUNCTIONS:</u>

Defining a Function

The general form of a function definition in C programming language is as follows −

```
return_type function_name( parameter list ) {

   body of the function

}
```

A function definition in C programming consists of a *function header* and a *function body*. Here are all the parts of a function −

- **Return Type** − A function may return a value. The **return_type** is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return_type is the keyword **void**.

- **Function Name** − This is the actual name of the function. The function name and the parameter list together constitute the function signature.

- **Parameters** − A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is referred to as actual parameter or argument. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.
- **Function Body** − The function body contains a collection of statements that define what the function does.

Function Declarations

A function **declaration** tells the compiler about a function name and how to call the function. The actual body of the function can be defined separately.

A function declaration has the following parts −

```
return_type function_name( parameter list );
```

Calling a Function

While creating a C function, you give a definition of what the function has to do. To use a function, you will have to call that function to perform the defined task.

When a program calls a function, the program control is transferred to the called function. A called function performs a defined task and when its return statement is executed or when its function-ending closing brace is reached, it returns the program control back to the main program.

To call a function, you simply need to pass the required parameters along with the function name, and if the function returns a value, then you can store the returned value.

- ## POINTERS:

A **pointer** is a variable whose value is the address of another variable, i.e., direct address of the memory location. Like any variable or constant, you must declare a pointer before using it to store any variable address. The general form of a pointer variable declaration is −

```
type *var-name;
```

How to Use Pointers?

There are a few important operations, which we will do with the help of pointers very frequently. **(a)** We define a pointer variable, **(b)** assign the address of a variable to a pointer and **(c)** finally access the value at the address available in the pointer variable. This is done by using unary operator ***** that returns the value of the variable located at the address specified by its operand. The following example makes use of these operations –

```c
#include <stdio.h>

int main () {

   int  var = 20;   /* actual variable declaration */
   int  *ip;        /* pointer variable declaration */

   ip = &var;  /* store address of var in pointer variable*/

   printf("Address of var variable: %x\n", &var  );

   /* address stored in pointer variable */
   printf("Address stored in ip variable: %x\n", ip );

   /* access the value using the pointer */
   printf("Value of *ip variable: %d\n", *ip );

   return 0;
```

- <u>STRUCTURES:</u>

Defining a Structure

To define a structure, you must use the **struct** statement. The struct statement defines a new data type, with more than one member. The format of the struct statement is as follows −

```
struct [structure tag] {


   member definition;

   member definition;

   ...

   member definition;

} [one or more structure variables];
```

The **structure tag** is optional and each member definition is a normal variable definition, such as int i; or float f; or any other valid variable definition. At the end of the structure's definition, before the final semicolon, you can specify one or more structure variables but it is optional.

Accessing Structure Members:

To access any member of a structure, we use the **member access operator (.)**. The member access operator is coded as a period between the structure variable name and the structure member that we wish to access. You would use the keyword **struct** to define variables of structure type.

STATUS:

We have studied the topics related to the project.

Week – 2

Tasks to be performed this week:

- To create login information
- To create mainmenu using switch case

Tasks performed this week:

- SECURITY:
  security() function is created so that only the admin can open the system by entering login id and password.

```c
void security()
{
    char username[10],password[10],ch;
    int i;
    system("cls");
    printf("\nEnter Username:\n");
    gets(username);
    printf("\nEnter Password:\n");
    for(i=0;i<6;i++)
    {
        ch=getch();
        password[i]=ch;
        ch='*';
        printf("%c",ch);
    }
    password[i]='\0';
    if(strcmp(username,"Rinki")==0 && strcmp(password,"redbus")==0)
    {
        printf("\nCongratulations..!!\nPress enter to continue..\n");
        getch();
        system("cls");
        mainmenu();
    }
    else
    {
        printf("\nWrong Username or Password..!\nTo try again press enter.\n");
        getch();
        system("cls");
        security();
    }
    getch();
}
```

- <u>MAINMENU</u>
  mainmenu() function is created to display the options available to the admin regarding the bus reservation changes.

```c
void mainmenu()
{
    system("color 3f");
    int j;
    printf("\n\n\t\t\tREDBUS.IN\n");
    printf("\t\t\tMY BUS....My seat\n");
    printf("1.Add new bus route\n");
    printf("2.See all available bus route\n");
    printf("3.Load buses to the route (Jpr to Delhi)\n");
    printf("4.See buses to the route (Jpr to Delhi)\n");
    printf("5.Delete Bus to the route.\n");
    printf("6.Load seats to the bus\n");
    printf("7.Book a ticket\n");
    printf("8.Exit\n");
    printf("Enter your choice\n");
    scanf("%d",&j);
    system("cls");
    system("color 1F");
    switch(j)
    {
        case 1:
            addNewBusRoute();
            break;
        case 2:
            displayAllBusRoute();
            break;
        case 3:
            loadBuses();
            break;
        case 4:
            displayBuses();
            break;
        case 5:
            deleteDetails();
            break;
        case 6:
            loadSeats();
            break;
        case 7:
            bookTicket();
            break;
        case 8:
            exit(0);
            break;
        default:
            system("cls");
            mainmenu();

    }
}
```

STATUS:

We have designed the security window and main menu.

Fig. SECURITY WINDOW:

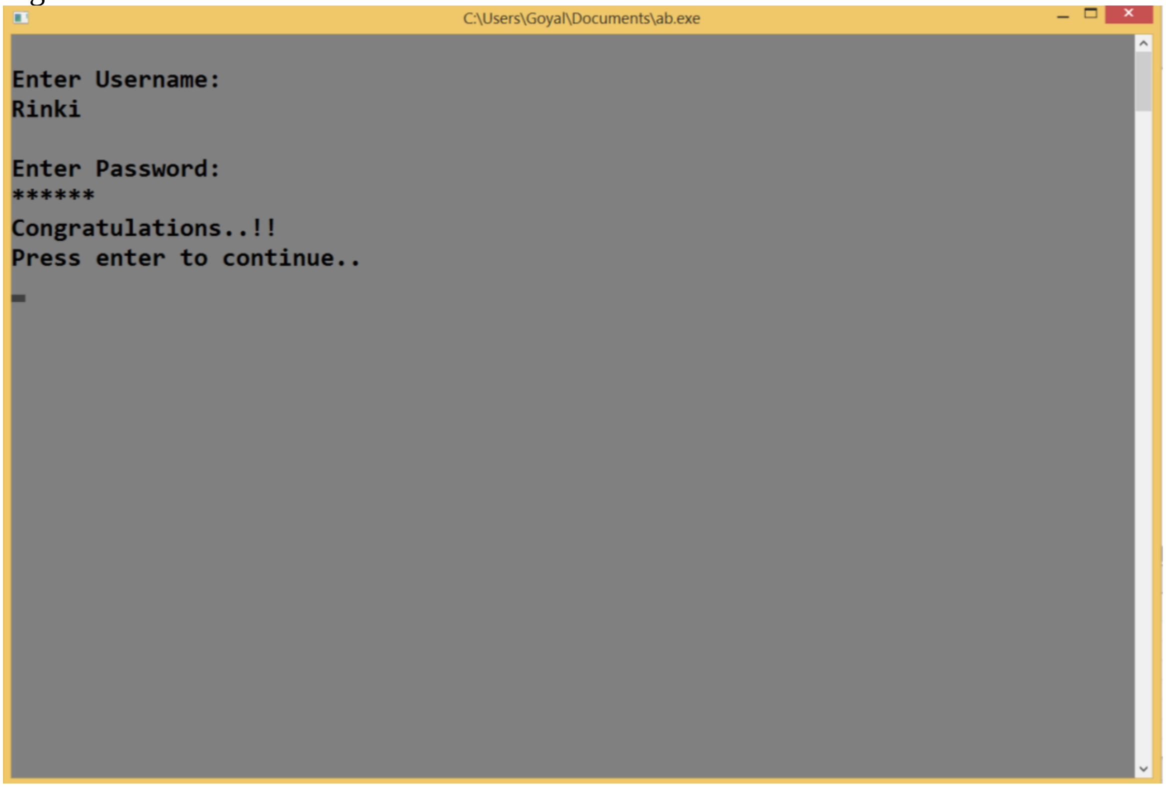
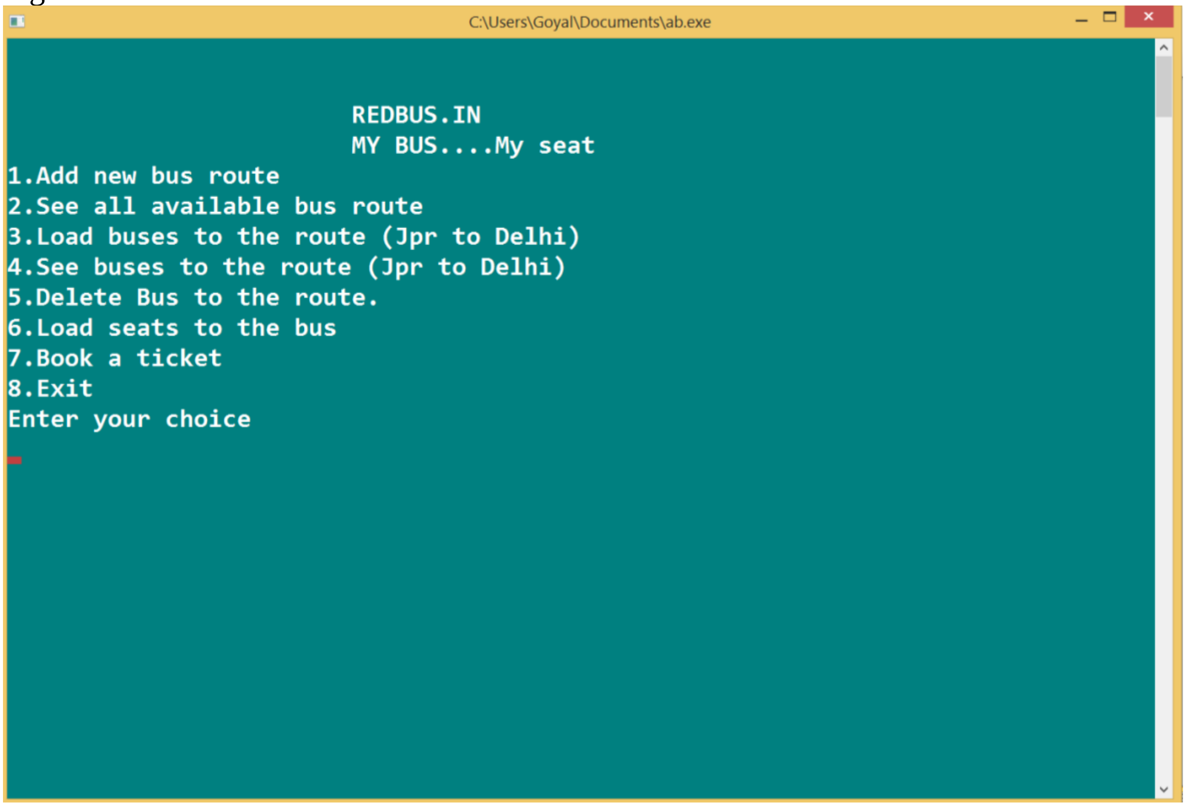


Fig. MAINMENU:



Week – 3

Tasks to be performed this week:

- To insert and display bus routes.
- To insert, delete and display buses to the route Jaipur to Delhi.

Tasks performed this week:

- INSERT BUS ROUTES:
  addNewBusRoute() function is used to add new routes to the system.

```
void addNewBusRoute()
{
int i;
FILE *fp;
fp=fopen("bus.txt","a");
struct checkpoint cp;
struct checkpoint *pptr;
pptr=&cp;
printf("Enter Bus route number\n");
scanf("%d",&pptr->no);
printf("Enter source\n");
scanf("%s",pptr->startpoint);
printf("Enter destination\n");
scanf("%s",pptr->endpoint);
fwrite(pptr,sizeof(*pptr),1,fp);
fclose(fp);
printf("Congratulations!!\nYour bus route is saved\n");
printf("\nDo you want to add more route?\n1.Yes\t\t\t2.No\n");
scanf("%d",&i);
switch(i)
{
    case 1:
        system("cls");
        addNewBusRoute();
        break;
    case 2:
        system("cls");
        mainmenu();
        break;
    default:
        system("cls");
        printf("Enter appropriate choice\n");
        addNewBusRoute();

}
}
```

- <u>DISPLAY BUS ROUTES:</u>

displayAllBusRoute() function is used to display the available Bus routes.

```c
void displayAllBusRoute()
{
    int i;
    int ch;
    FILE *fp1;
    struct checkpoint cp;
    struct checkpoint *pptr;
    pptr=&cp;
    fp1=fopen("bus.txt","r");

        printf("Available Bus Routes are:\n");
        printf("Route No.     Starting Point        Destination");
        printf("\n-----------------------------------------------------------\n");
        while(fread(pptr,sizeof(*pptr),1,fp1))
        {
            printf("%d               %s                 %s\n",pptr->no,pptr->startpoint,pptr->endpoint);
        }
        printf("\n1.Get directed to main menu.\n2.Book a ticket.\n3.Exit\n");
        printf("Enter your choice\n");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:
                system("cls");
                mainmenu();
                break;
            case 2:
                system("cls");
                bookTicket();
                break;
            case 3:
                exit(0);
                break;
            default:
                printf("\nEnter appropriate choice\n");
                displayAllBusRoute();
    }

    fclose(fp1);
}
```

- <u>INSERT BUSES TO THE ROUTE:</u>

  loadBuses() function is used to add buses to the route Jaipur to Delhi.

```c
void loadBuses()
{
    int i;
    FILE *fp;
    fp = fopen("busdetail.txt","a");
    struct bus bs;
    struct bus *ptr;
    ptr=&bs;
    printf("Enter bus number\n");
    scanf("%d",&ptr->busno);
    printf("Enter bus name\n");
    scanf("%s",ptr->busname);
    printf("Enter bus arrival time(hh:mm)\n");
    scanf("%s",ptr->arrtime);
    printf("Enter bus departure time(hh:mm)\n");
    scanf("%s",ptr->deptime);
    printf("Enter time duration\n");
    scanf("%s",ptr->duration);
    printf("Enter bus status(AC / Non-AC)\n");
    scanf("%s",ptr->status);
    printf("Enter bus fare\n");
    scanf("%d",&ptr->fare);
    fwrite(ptr,sizeof(*ptr),1,fp);
    fclose(fp);
    printf("Congratulations!!\nYour bus details are saved\n");
    printf("\nDo you want to add more buses to the route?\n1.Yes\t\t\t2.No\n");
    scanf("%d",&i);
    switch(i)
    {
        case 1:
            system("cls");
            loadBuses();
            break;
        case 2:
            system("cls");
            mainmenu();
            break;
        default:
            system("cls");
            printf("\nEnter appropriate choice");
            loadBuses();
    }
}
```

- <u>DISPLAY BUSES TO THE ROUTE:</u>
  displayBuses() function is used to display the buses available to the route Jaipur to Delhi.

```c
void displayBuses()
{
    int i;
    int ch;
    FILE *fp1;
    struct bus bs;
    struct bus *ptr;
    ptr=&bs;
    fp1=fopen("busdetail.txt","r");

        printf("Available Buses to the route are:\n\n");
        printf("Bus No.  Bus Name      Arrival Time   Departure Time   Duration   Fare   Status--------------------------------------------------------------------------------\n");
        while(fread(ptr,sizeof(*ptr),1,fp1))
        {
        printf(" %d      %s      %s         %s           %s      %d      %s\n",ptr->busno,ptr->busname,ptr->arrtime,ptr->deptime,ptr->duration,ptr->fare,ptr->status);
        }
        printf("\n1.Get directed to main menu.\n2.Book a ticket.\n3.Exit\n");
        printf("Enter your choice\n");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:
                system("cls");
                mainmenu();
                break;
            case 2:
                system("cls");
                bookTicket();
                break;
            case 3:
                exit(0);
            default:
                printf("\nEnter appropriate choice\n");
                displayBuses();
        }
}
```

- <u>DELETE BUSES TO THE ROUTE:</u>
  deleteDetails() function is used to delete a bus to the given route.

```c
void deleteDetails()
{
    FILE *fp;
    FILE *fpt;
    struct bus bs;
    int r,a;
    printf("Enter the Bus no you want to delete :");
    scanf("%d",&r);
    {
        fp=fopen("busdetail.txt","rb+");
        fpt=fopen("TempFile.txt","wb+");
        while(fread(&bs,sizeof(bus),1,fp))
        {
            a=bs.busno;
            if(a!=r)
            fwrite(&bs,sizeof(bus),1,fpt);
        }
        fclose(fp);
        fclose(fpt);
        fp=fopen("busdetail.txt","wb+");
        fpt=fopen("TempFile.txt","rb+");
        while(fread(&bs,sizeof(bus),1,fpt))
        fwrite(&bs,sizeof(bus),1,fp);
        printf("\nRECORD DELETE\n");
    }
    fclose(fp);
    fclose(fpt);
}
```

<u>STATUS:</u>

We have designed the routes and bus details windows.

Fig. Insert Bus Route Window

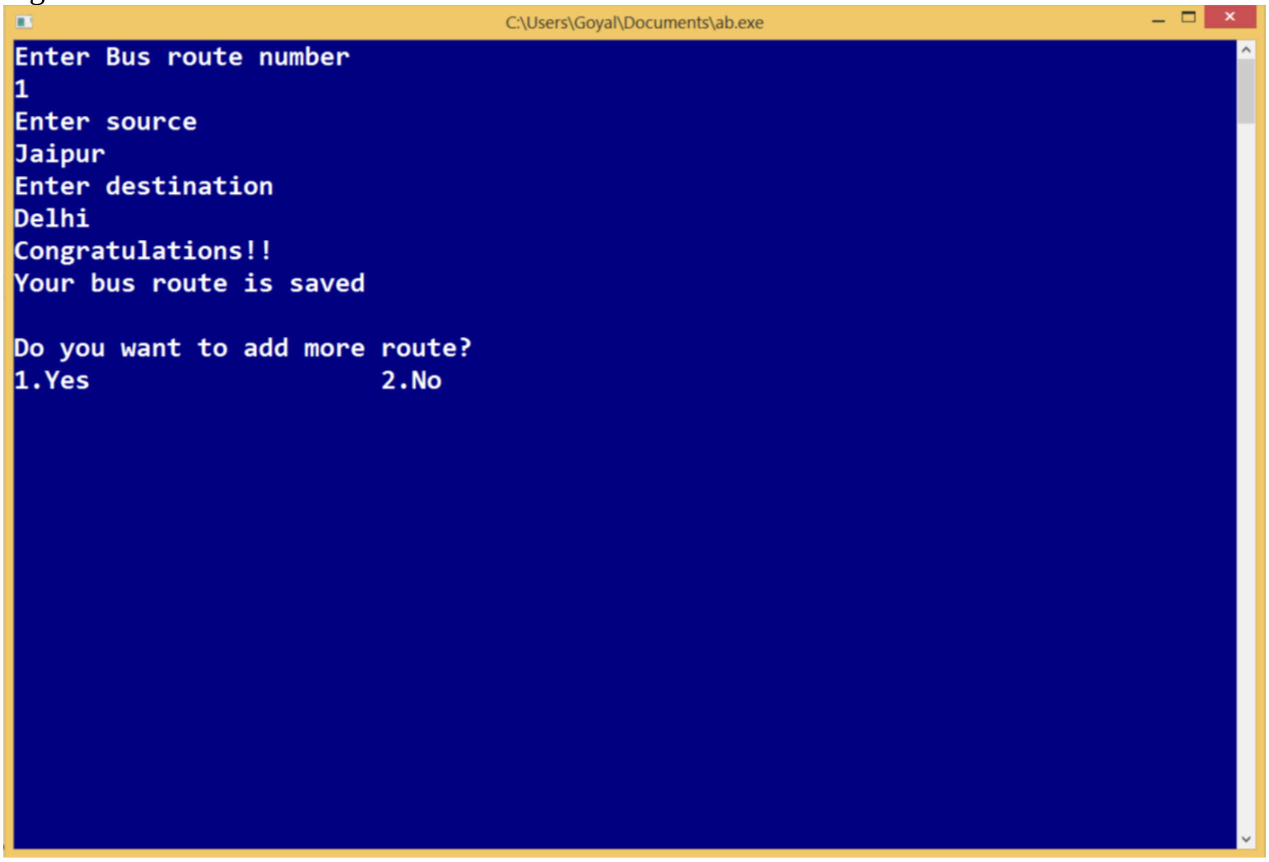


Fig. Display Available Bus Route

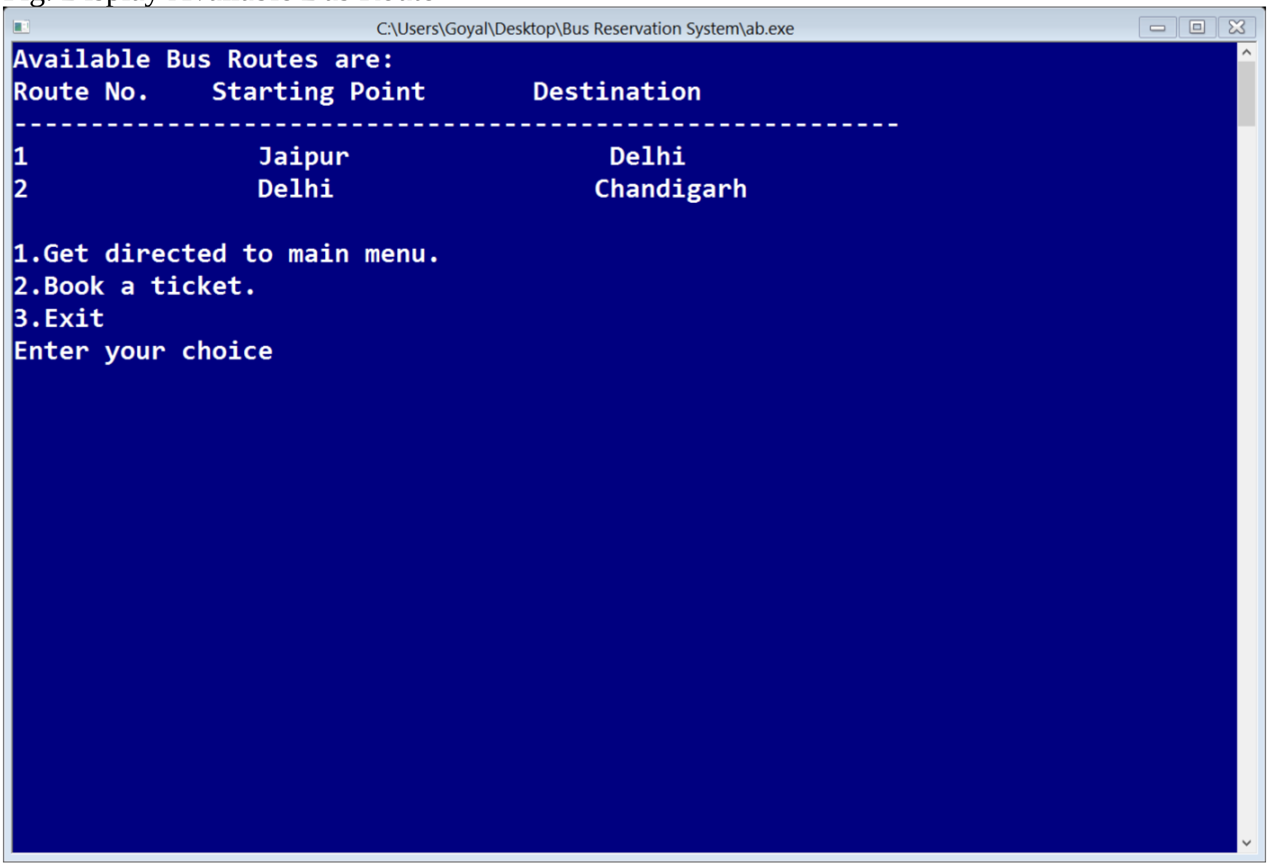


Fig. Insert Bus Detail Window

```
C:\Users\Goyal\Documents\ab.exe
Enter bus number
1
Enter bus name
Mahalaxmi
Enter bus arrival time(hh:mm)
10:30
Enter bus departure time(hh:mm)
23:30
Enter time duration
13:00
Enter bus status(AC / Non-AC)
AC
Enter bus fare
700
Congratulations!!
Your bus details are saved

Do you want to add more buses to the route?
1.Yes                   2.No
```

Fig. Available Buses Window

```
C:\Users\Goyal\Desktop\Bus Reservation System\ab.exe
Available Buses to the route are:

Bus No.  Bus Name       Arrival Time   Departure Time   Duration   Fare   Status
--------------------------------------------------------------------------------

 1       Mahalaxmi      08:30          22:30            14:00      800      AC
 2       Kalpana      13:30          19:30            06:00      800      AC

1.Get directed to main menu.
2.Book a ticket.
3.Exit
Enter your choice
```

Week – 4

Tasks to be performed this week:

- To load seats to a bus.
- To book tickets in a bus.

Tasks performed this week:

- LOAD SEATS TO A BUS:
  To load seats to a bus and initially make the status unbooked, loadBuses() function is used.

```c
void loadSeats()
{
    system("cls");
    int i,f;
    int ch;
    FILE *fp1;
    struct bus bs;
    struct bus *ptr;
    ptr=&bs;
    fp1=fopen("busdetail.txt","rb+");

    printf("Available Buses to the route are:\n");
    printf("Bus No.  Bus Name      Arrival Time   Departure Time   Duration   Fare   Status\n--------------------------------------------------------------------------------\n");
    while(fread(ptr,sizeof(*ptr),1,fp1))
    {
        printf(" %d      %s      %s          %s            %s      %d       %s\n",ptr->busno,ptr->busname,ptr->arrtime,ptr->deptime,ptr->duration,ptr->fare,ptr->status);
    }
    printf("Which bus would you like to load seats to(Enter Bus no.)\n");
    scanf("%d",&f);

    switch(f)
    {
        case 1:
            loadMahalaxmiSeats();
            break;
        case 3:
            //saraswati();
            break;
        default:
            printf("Select correct option\n");
            loadSeats();
    }
}
```

- <u>BOOK A TICKET:</u>

  The following functions are used to book a ticket.

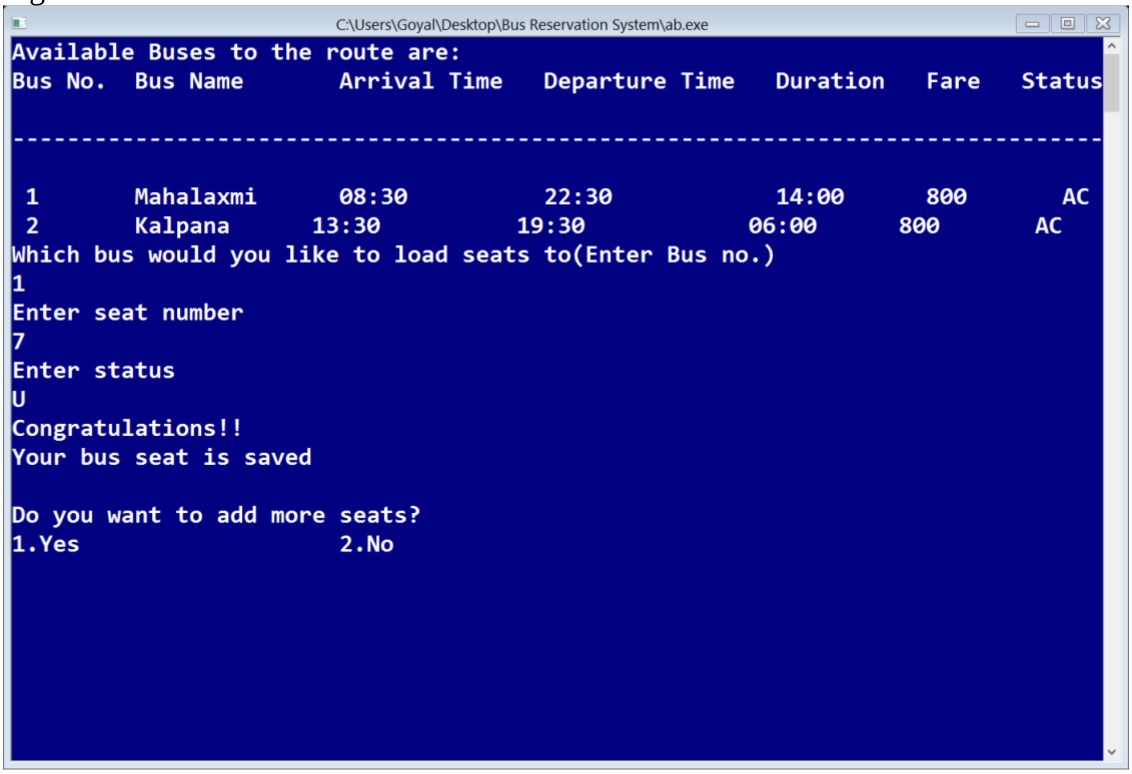
```c
void bookTicket()
{
    char source[10],des[10];
    printf("\t\t Book a Ticket \t");
    int i;
    int ch;
    FILE *fp1;
    struct checkpoint cp;
    struct checkpoint *pptr;
    pptr=&cp;
    fp1=fopen("bus.txt","r");

        printf("Available Bus Routes are:\n");
        while(fread(pptr,sizeof(*pptr),1,fp1))
        {
            printf("%d  %s  %s\n",pptr->nc,pptr->startpoint,pptr->endpoint);
        }
    printf("\nEnter source\n");
    scanf("%s",source);
    printf("\nEnter destination\n");
    scanf("%s",des);
    if((strcmp(source,"jaipur")==0||strcmp(source,"jpr")==0||strcmp(source,"Jaipur")==0) && (strcmp(des,"Delhi")==0||strcmp(des,"delhi")==0||strcmp(des,"dlh")==0))
    {
        system("cls");
        int i,f;
        int ch;
        FILE *fp1;
        struct bus bs;
        struct bus *ptr;
        ptr=&bs;
        fp1=fopen("busdetail.txt","rb+");

        printf("Available Buses to the route are:\n");
        printf("Bus No.  Bus Name      Arrival Time   Departure Time   Duration   Fare   Status\n-----------------------------------------------------------------------------\n");
        while(fread(ptr,sizeof(*ptr),1,fp1))
        {
            printf(" %d      %s      %s         %s           %s      %d       %s\n",ptr->busno,ptr->busname,ptr->arrtime,ptr->deptime,ptr->duration,ptr->fare,ptr->status);
        }

        printf("Which bus would you like to select(Enter Bus no.)\n");
        scanf("%d",&f);

        switch(f)
        {
            case 1:
                mahalaxmi();
                break;
            case 3:
                //saraswati();
                break;
            default:
                printf("Select correct option\n");
                bookTicket();

        }


    }
    else
    {
        system("cls");
        printf("\n No available buses to this route.");
        mainmenu();
    }
}
```

```c
void mahalaxmi()
{
    int i;
    int m;
    char g;
    FILE *mbc;
    struct mahaseat mseat;
    mbc=fopen("Mahalaxmi.txt","rb+");
    while(fread(&mseat,sizeof(struct mahaseat),1,mbc)==1)
    {
            printf("\t%d\t%c\n",mseat.seatno,mseat.status1);
    }
    fclose(mbc);
    printf("\nHow many tickets you want to book?\n");
    scanf("%d",&m);
    printf("\nPlease enter seat number.\n");
    int *p = (int *)malloc(sizeof(int)*(m));
    for(i=0;i<m;i++)
    {
        scanf("%d",&p[i]);
    }
    printf("Your seat numbers are:\n");
    for(i=0;i<m;i++)
    {
        printf("%d\n",p[i]);
    }
    getch();
    checkingMahalaxmi(m,p);
    FILE *mbcf;
    mbcf = fopen("Mahalaxmi.txt","rb+");
    struct mahaseat upstatus;
    while(fread(&upstatus,sizeof(struct mahaseat),1,mbcf)==1)
    {
        for(i=0;i<m;i++)
        {
            if(upstatus.seatno==p[i])
            {
                fseek(mbcf,-sizeof(upstatus),SEEK_CUR);
                upstatus.status1='B';
                fwrite(&upstatus,sizeof(upstatus),1,mbcf);
                fseek(mbcf,0,SEEK_CUR);
                printf("\t%d %c\n",upstatus.seatno,upstatus.status1);
                break;
            }
        }
    }
    printf("CONGRATS!! You have booked your tickets..\n");
    fclose(mbcf);
    getch();
    system("cls");
    mainmenu();
}
```

STATUS:

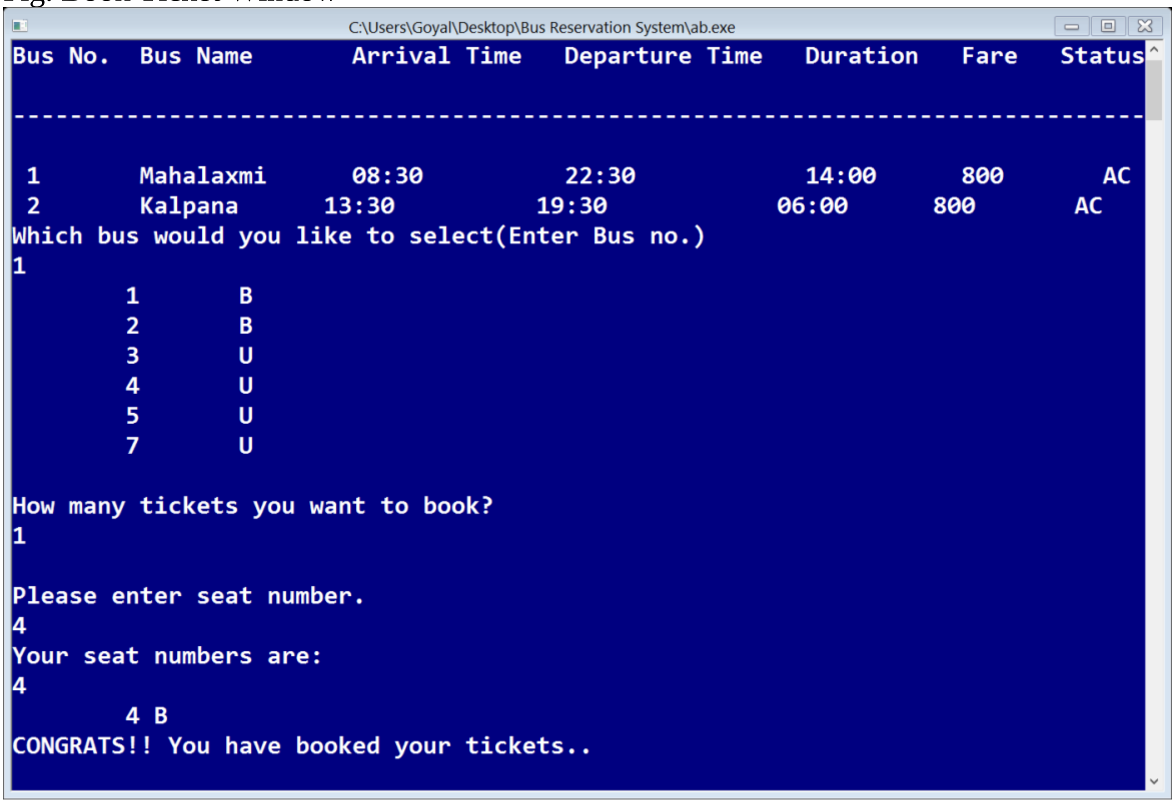
We have designed the load seats and book ticket windows.

Fig. Load Seats



```
C:\Users\Goyal\Desktop\Bus Reservation System\ab.exe
Available Buses to the route are:
Bus No.  Bus Name       Arrival Time   Departure Time   Duration   Fare   Status
--------------------------------------------------------------------------------
 1       Mahalaxmi      08:30          22:30            14:00      800      AC
 2       Kalpana      13:30          19:30            06:00      800      AC
Which bus would you like to load seats to(Enter Bus no.)
1
Enter seat number
7
Enter status
U
Congratulations!!
Your bus seat is saved

Do you want to add more seats?
1.Yes                   2.No
```

Fig. Book Ticket Window



```
C:\Users\Goyal\Desktop\Bus Reservation System\ab.exe
Bus No.  Bus Name       Arrival Time   Departure Time   Duration   Fare   Status
--------------------------------------------------------------------------------
 1       Mahalaxmi      08:30          22:30            14:00      800      AC
 2       Kalpana      13:30          19:30            06:00      800      AC
Which bus would you like to select(Enter Bus no.)
1
        1       B
        2       B
        3       U
        4       U
        5       U
        7       U

How many tickets you want to book?
1

Please enter seat number.
4
Your seat numbers are:
4
        4 B
CONGRATS!! You have booked your tickets..
```