AIRLINE BOOKING

ANUSHTHA SINGH KUSHWAH

October 2023

1 About This Project

Air-Line Booking Project is a mini project which let user Book there Desired Seat This Project is Completely Build on C Program

Let's See the working of tis project with an Example.
Working: Supose Anushtha is Going on a vacation and She Needs to Book her Desired Seat For that she will interface with this Program in which First she will Encounter 5 options

```
**********************************************

**  Welcome to Anushtha's airline system  **

**********************************************


 Please enter your choice from below (1-4):

 1. Reservation

 2. Cancel

 3. DISPLAY RECORDS

 4. Explore Some Famous Destinations

 5. EXIT

 Enter your choice:
```

Now that she want to reserve her seat She will choose option 1

```
 Enter your choice: 1

Available Seats: A-1 A-2 A-3 A-4 A-5 A-6 A-7 A-8 A-9 A-10 A-11 A-12 A-13 A-14 A-15
Enter the seat number you want to reserve (1-15): 5
Enter passport number (5 characters): 34344
Enter name (up to 14 characters): Anushtha
Enter email address (up to 14 characters): anush@gmail.com
Enter destination (up to 14 characters): Japan
Seat booking successful! Your seat number is: Seat A-5
```

After Selecting option for reservation the program will show her seats which are Available for booking
After Selecting her seat(5) She has to Enter her Basic Details
Details

1. Passport Number
2. Name
3. Email ID
4. Destination

After Filling the Details the program will show her Seat number now we can see her details by clicking on the Third option Display Records

```
 Please enter your choice from below (1-4):

 1. Reservation

 2. Cancel

 3. DISPLAY RECORDS

 4. Explore Some Famous Destinations

 5. EXIT

 Enter your choice: 3
Seat    Passport        Name                    Email                   Destination
A-5     34344           Anushtha                anush@gmail.com         Japan
```

Now the is another Customer Khushi who is also going in same flight so she will also book her Seat by filling out Details

```
Enter your choice: 1

Available Seats: A-1 A-2 A-3 A-4 A-6 A-7 A-8 A-9 A-10 A-11 A-12 A-13 A-14 A-15
Enter the seat number you want to reserve (1-15): 2
Enter passport number (5 characters): 23233
Enter name (up to 14 characters): Khushi
Enter email address (up to 14 characters): khushi@gmail.com
Enter destination (up to 14 characters): Japan
Seat booking successful! Your seat number is: Seat A-2
```

But this time Seat number 5 is not available so khushi will not see the A-5 Seat in Available seats.
Now when we see the Records We can see the Details Filled by Both Anushtha and Khushi

```
Please enter your choice from below (1-4):

1. Reservation

2. Cancel

3. DISPLAY RECORDS

4. Explore Some Famous Destinations

5. EXIT

Enter your choice: 3
Seat    Passport        Name                    Email                   Destination
A-2     23233           Khushi                  khushi@gmail.com                Japan
A-5     34344           Anushtha                anush@gmail.com         Japan
```

But Khushi Changed her Plan She wanted to find any other place to visit for that first she will cancel her Reservation by clicking on Second option

```
Please enter your choice from below (1-4):

1. Reservation

2. Cancel

3. DISPLAY RECORDS

4. Explore Some Famous Destinations

5. EXIT

Enter your choice: 2
Enter passport number to delete record: 23233
Booking has been deleted.


Please enter your choice from below (1-4):

1. Reservation

2. Cancel

3. DISPLAY RECORDS

4. Explore Some Famous Destinations

5. EXIT

Enter your choice: 3
Seat    Passport        Name                    Email                   Destination
A-5     34344           Anushtha                anush@gmail.com         Japan
```

Now in Display Records Details for Khushi Got Deleted and Since she wanted to explore any other Destination she can Select option forth and check for famous destination.

```
 Please enter your choice from below (1-4):

 1. Reservation

 2. Cancel

 3. DISPLAY RECORDS

 4. Explore Some Famous Destinations

 5. EXIT

 Enter your choice: 4
**************************
* Enter Your Prefernce *
* 1. With Family       *
* 2. With Friends      *
* 3. Solo              *
* 4. With Partner      *
**************************
Choose From 1-4 :3
*****************
*Japan          *
*Udaipur        *
*Malaysia       *
*Costa Rica     *
*Australia      *
*Singapore      *
*Spain          *
*Hampi          *
*Amsterdam      *
*Varanasi       *
*Manali         *
*Kasol          *
*Varkala        *
*Darjeeling     *
*Varkala        *
*****************
```

Debbugging

```
PS C:\Users\anush\Downloads\Airline_Booking_In_C_With_Source_Code (1)\AirlineBooking-main> gcc fir.c -g -o fir
PS C:\Users\anush\Downloads\Airline_Booking_In_C_With_Source_Code (1)\AirlineBooking-main> gdb ./fir
GNU gdb (GDB) 7.6.1
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "mingw32".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from C:\Users\anush\Downloads\Airline_Booking_In_C_With_Source_Code (1)\AirlineBooking-main\fir.exe...done.
(gdb) list
29                  printf("A-%d ", i + 1);
30              }
31          }
32          printf("\n");
33      }
34
35      int main() {
36          int choice;
37
38          printf("\n\n*****************************************");
(gdb) r
Starting program: C:\Users\anush\Downloads\Airline_Booking_In_C_With_Source_Code (1)\AirlineBooking-main/./fir.exe
[New Thread 8932.0xaac]
[New Thread 8932.0x2b34]


*****************************************

**  Welcome to Anushtha's airline system  **

*****************************************


 Please enter your choice from below (1-4):

 1. Reservation

 2. Cancel

 3. DISPLAY RECORDS

 4. Explore Some Famous Destinations

 5. EXIT
```

Profilling

```
Flat profile:

Each sample counts as 0.01 seconds.
 no time accumulated

  %   cumulative   self              self     total
 time   seconds   seconds    calls  Ts/call  Ts/call  name
  0.00      0.00     0.00        1     0.00     0.00  savefile

 %         the percentage of the total running time of the
time       program used by this function.

cumulative a running sum of the number of seconds accounted
 seconds   for by this function and those listed above it.

 self      the number of seconds accounted for by this
seconds    function alone.  This is the major sort for this
           listing.

calls      the number of times this function was invoked, if
           this function is profiled, else blank.

 self      the average number of milliseconds spent in this
ms/call    function per call, if this function is profiled,
           else blank.

 total     the average number of milliseconds spent in this
ms/call    function and its descendents per call, if this
           function is profiled, else blank.

name       the name of the function.  This is the minor sort
           for this listing. The index shows the location of
           the function in the gprof listing. If the index is
           in parenthesis it shows where it would appear in
           the gprof listing if it were to be printed.
```

```
Copyright (C) 2012-2017 Free Software Foundation, Inc.

Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.


                     Call graph (explanation follows)


granularity: each sample hit covers 4 byte(s) no time propagated

index % time    self  children    called     name
                0.00    0.00       1/1           main [85]
[2]      0.0    0.00    0.00       1         savefile [2]
-----------------------------------------------

 This table describes the call tree of the program, and was sorted by
 the total amount of time spent in each function and its children.

 Each entry in this table consists of several lines.  The line with the
 index number at the left hand margin lists the current function.
 The lines above it list the functions that called this function,
 and the lines below it list the functions this one called.
 This line lists:
     index      A unique number given to each element of the table.
                Index numbers are sorted numerically.
                The index number is printed next to every function name so
                it is easier to look up where the function is in the table.

     % time     This is the percentage of the `total' time that was spent
                in this function and its children.  Note that due to
                different viewpoints, functions excluded by options, etc,
                these numbers will NOT add up to 100%.

     self       This is the total amount of time spent in this function.

     children   This is the total amount of time propagated into this
                function by its children.
```

```
    called     This is the number of times the function was called.
               If the function called itself recursively, the number
               only includes non-recursive calls, and is followed by
               a `+' and the number of recursive calls.

    name       The name of the current function.  The index number is
               printed after it.  If the function is a member of a
               cycle, the cycle number is printed between the
               function's name and the index number.


For the function's parents, the fields have the following meanings:

    self       This is the amount of time that was propagated directly
               from the function into this parent.

    children   This is the amount of time that was propagated from
               the function's children into this parent.

    called     This is the number of times this parent called the
               function `/' the total number of times the function
               was called.  Recursive calls to the function are not
               included in the number after the `/'.

    name       This is the name of the parent.  The parent's index
               number is printed after it.  If the parent is a
               member of a cycle, the cycle number is printed between
               the name and the index number.

If the parents of the function cannot be determined, the word
`<spontaneous>' is printed in the `name' field, and all the other
fields are blank.

For the function's children, the fields have the following meanings:

    self       This is the amount of time that was propagated directly
               from the child into the function.
```

```
     children   This is the amount of time that was propagated from the
                child's children to the function.

     called     This is the number of times the function called
                this child `/' the total number of times the child
                was called.  Recursive calls by the child are not
                listed in the number after the `/'.

     name       This is the name of the child.  The child's index
                number is printed after it.  If the child is a
                member of a cycle, the cycle number is printed
                between the name and the index number.

 If there are any cycles (circles) in the call graph, there is an
 entry for the cycle-as-a-whole.  This entry shows who called the
 cycle (as parents) and the members of the cycle (as children.)
 The `+' recursive calls entry shows the number of function calls that
 were internal to the cycle, and the calls entry for each member shows,
 for that member, how many times it was called from other members of
 the cycle.


Copyright (C) 2012-2017 Free Software Foundation, Inc.

Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.


Index by function name

   [2] savefile
```