San José State
UNIVERSITY

A REPORT ON CMPE-207 PROJECT

"DISTRIBUTED AIRLINE RESERVATION SYSTEM"

PROJECT GUIDE

Dr. Rod Fatoohi

Project Group - 10

Project Associates

Suhas Janardhan(011169054)

Nikhil Vijayakumar Kengalahalli(011418069)

Lavanya Bale(011467157)

CONTENTS

ABSTRACT

The main objective of the project is to design a client/server model based Distributed Airline Reservation System application, which will enable the customers to search and book flights.

The project mainly consists of three main players.

- First is the System Admin who can add or delete Airline Representatives.
- Second is the Airline Representative who can add/delete specific flights, view their flights.
- And the customers who can view/book/delete the flight tickets.

The report contains the details of design and implementation carried out during the development of the Distributed Airline Reservation System Project.

.

INTRODUCTION

Airline Reservation System allows the user to enter their query to search for flight and reserve their flight ticket. It allows users to book their tickets from any location, thus saving their time and effort. Users can place their query as per their requirement and get results immediately on particular flight as per their desired location and as per entered price.

This project is aimed at developing Distributive Airline Reservation System Project. An airline reservation system will help the user to book their flight tickets without having to go to a booking counter to book a ticket, hence saving their time and money. This Reservation System comes with the ease of accessibility. It can be accessed by the user at any time and from any location. Airline Reservation System Project will be user friendly and providing all the information based on the user requirements.

There is one centralized server whose main objective is to authenticate the users. Once the user enters his or her credentials server checks for the matching credentials in the database and sends the success or failure message accordingly.

To book a ticket, users have to sign in using login menu. If a user has not registered, he can sign up by creating a new username and password. If he has already registered, he can login by entering his already existing username and password to gain access to the system. After login user will be provided with the information on screen, which will provide information on the flights. The customer can search for flight based on the information displayed and book a ticket or cancel it. Customer can search flights by entering necessary information such as email id, departure location, arrival location, number of seats etc. Based on the information taken from the customer, flights are searched in the database. All the flight information is stored in MySQL database. Only those flights which match the requirements of the customers are displayed. The customer can accordingly book his ticket by selecting the desired flight.

After booking ticket, the customer will receive a confirmation mail with the flight information, from the respective airlines.

KEY TERMS AND CONCEPTS

- **Client and server role**

  The Client-server characteristic describes the relationship of cooperating programs in an application. The server component provides a function or service to one or many clients, which initiate requests for such services.

  Servers are classified by the services they provide. For instance, a web server serves web pages and a file server serves computer files. A shared resource may be any of the server computer's software and electronic components, from programs and data to processors and storage devices. The sharing of resources of a server constitutes a service.

  Whether a computer is a client, a server, or both, is determined by the nature of the application that requires the service functions. For example, a single computer can run web server and file server software at the same time to serve different data to clients making different kinds of requests. Client software can also communicate with server software within the same computer. Communication between servers, such as to synchronize data, is sometimes called inter-server or server-to-server communication.

- **Client and server communication**

  In general, a service is an abstraction of computer resources and a client does not have to be concerned with how the server performs while fulfilling the request and delivering the response. The client only has to understand the response based on the well-known application protocol, i.e. the content and the formatting of the data for the requested service.

  Clients and servers exchange messages in a request–response messaging pattern: The client sends a request, and the server returns a response. This exchange of messages is an example of inter-process communication. To communicate, the computers must have a common language, and they must follow rules so that both the client and the server know what to expect. The language and rules of communication are defined in a communications protocol. All client-server protocols operate in the application layer. The application-layer protocol defines the basic patterns of the dialogue. To formalize the data exchange even further, the server may implement an API (such as a web service). The API is an abstraction layer for such resources as databases and custom software. By restricting communication to a specific content format, it facilitates parsing. By abstracting access, it facilitates cross-platform data exchange.

  A server may receive requests from many different clients in a very short period of time. Because the computer can perform a limited number of tasks at any moment, it relies on a scheduling system to prioritize incoming requests from clients in order to accommodate them all in turn. To prevent abuse and maximize uptime, the server's software limits how a client can use the server's resources. Even so, a server is not immune from abuse. A denial of service attack exploits a server's obligation to process requests by bombarding it with requests incessantly. This inhibits the server's ability to respond to legitimate requests that can make the communication of web easier.

- **TCP IP**

TCP/IP is a two-layer program. The higher layer, Transmission Control Protocol, manages the assembling of a message or file into smaller packets that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, Internet Protocol, handles the address part of each packet so that it gets to the right destination. Each gateway computer on the network checks this address to see where to forward the message. Even though some packets from the same message are routed differently than others, they'll be reassembled at the destination.
TCP/IP uses the client/server model of communication in which a computer user (a client) requests and is provided a service (such as sending a Web page) by another computer (a server) in the network. TCP/IP communication is primarily point-to-point, meaning each communication is from one point (or host computer) in the network to another point or host computer. TCP/IP and the higher-level applications that use it are collectively said to be "stateless" because each client request is considered a new request unrelated to any previous one. Being stateless frees network paths so that everyone can use them continuously.

- **Database**

A database is an organized collection of data. It is the collection of schemas, tables, queries, reports, views, and other objects. The data are typically organized to model aspects of reality in a way that supports processes requiring information, such as modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

A database management system (DBMS) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyze data. A general-purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases. Well-known DBMSs include MySQL, Microsoft SQL Server, Oracle, Sybase, SAP HANA, and IBM DB2. A database is not generally portable across different DBMSs, but different DBMS can interoperate by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one DBMS. Sometimes a DBMS is loosely referred to as a 'database'.

In this project, we use MySQL to store all the information. The information is stored in the form of tables.

- **BSD Sockets**

Berkeley sockets is an application programming interface (API) for Internet sockets and Unix domain sockets, used for inter-process communication (IPC). It is commonly implemented as a library of linkable modules. It originated with the 4.2BSD Unix released in 1983.

A socket is an abstract representation (handle) for the local endpoint of a network communication path. The Berkeley sockets API represents it as a file descriptor (file handle) in the Unix philosophy that provides a common interface for input and output to streams of data. The BSD sockets API is written in the C programming language. Most other programming languages provide similar interfaces, typically written as a wrapper library based on the C API.

Few of the important socket API functions are listed below:

- **socket()** creates a new socket of a certain socket type, identified by an integer number, and allocates system resources to it.
- **bind()** is typically used on the server side, and associates a socket with a socket address structure, i.e. a specified local port number and IP address.
- **listen()** is used on the server side, and causes a bound TCP socket to enter listening state.
- **connect()** is used on the client side, and assigns a free local port number to a socket. In case of a TCP socket, it causes an attempt to establish a new TCP connection.
- **accept()** is used on the server side. It accepts a received incoming attempt to create a new TCP connection from the remote client, and creates a new socket associated with the socket address pair of this connection.
- **send()** and **recv(),** or **write()** and **read()**, or **sendto()** and **recvfrom()**, are used for sending and receiving data to/from a remote socket.
- **close()** causes the system to release resources allocated to a socket. In case of TCP, the connection is terminated.

FEATURES

1. IPv6

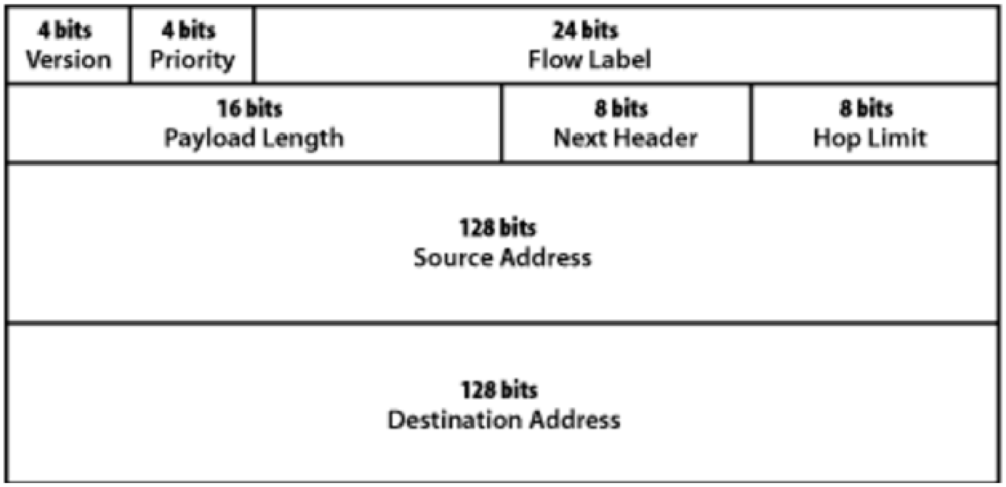
Internet Protocol version 6 (IPv6) is the most recent version of the Internet Protocol (IP), the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet. Every device on the Internet is assigned a unique IP address for identification and location definition. With the rapid growth of the Internet after commercialization in the 1990s, it became evident that far more addresses would be needed to connect devices than the IPv4 address space had available. IPv6 was developed by the Internet Engineering Task Force (IETF) to deal with the long-anticipated problem of IPv4 address exhaustion. IPv6 is intended to replace IPv4.

Comparison of IPv6 and IPv4:

IPv6 is extended version of IPv4 and has lot of advantages over IPv4.

- The main advantage of IPv6 over IPv4 is its larger address space. The length of an IPv6 address is 128 bits, compared with 32 bits in IPv4. The address space therefore has 2^{128} or approximately 3.4×10^{38} addresses.

IPv6 header is as shown below:



4 bits Version	4 bits Priority	24 bits Flow Label
16 bits Payload Length	8 bits Next Header	8 bits Hop Limit
128 bits Source Address		
128 bits Destination Address		

Fig1: *IPv6 Header structure*

The 128 bits of an IPv6 address are represented in 8 groups of 16 bits each. Each group is written as four hexadecimal digits and the groups are separated by colons (:). An example of this representation is **2001:0db8:0000:0000:0000:ff00:0042:8329.**

Listed below are some of the advantages of IPv6 over IPv4:

- **More Efficient Routing**

  IPv6 reduces the size of routing tables and makes routing more efficient and hierarchical. IPv6 allows ISPs to aggregate the prefixes of their customers' networks into a single prefix and announce this one prefix to the IPv6 Internet. In addition, in IPv6 networks, fragmentation is handled by the source device, rather than the router, using a protocol for discovery of the path's maximum transmission unit (MTU).

- **More Efficient Packet Processing**

  IPv6's simplified packet header makes packet processing more efficient. Compared with IPv4, IPv6 contains no IP-level checksum, so the checksum does not need to be recalculated at every router hop. Getting rid of the IP-level checksum was possible because most link-layer technologies already contain checksum and error-control capabilities. In addition, most transport layers, which handle end-to-end connectivity, have a checksum that enables error detection.

- **Directed Data Flows**

  IPv6 supports multicast rather than broadcast. Multicast allows bandwidth-intensive packet flows (like multimedia streams) to be sent to multiple destinations simultaneously, saving network bandwidth. Disinterested hosts no longer must process broadcast packets. In addition, the IPv6 header has a new field, named Flow Label, that can identify packets belonging to the same flow.

- **Simplified Network Configuration**

  Address auto-configuration (address assignment) is built in to IPv6. A router will send the prefix of the local link in its router advertisements. A host can generate its own IP address by appending its link-layer (MAC) address, converted into Extended Universal Identifier (EUI) 64-bit format, to the 64 bits of the local link prefix.

- **Support for New Services**

  By eliminating Network Address Translation (NAT), true end-to-end connectivity at the IP layer is restored, enabling new and valuable services. Peer-to-peer networks are easier to create and maintain, and services such as VoIP and Quality of Service (QoS) become more robust.

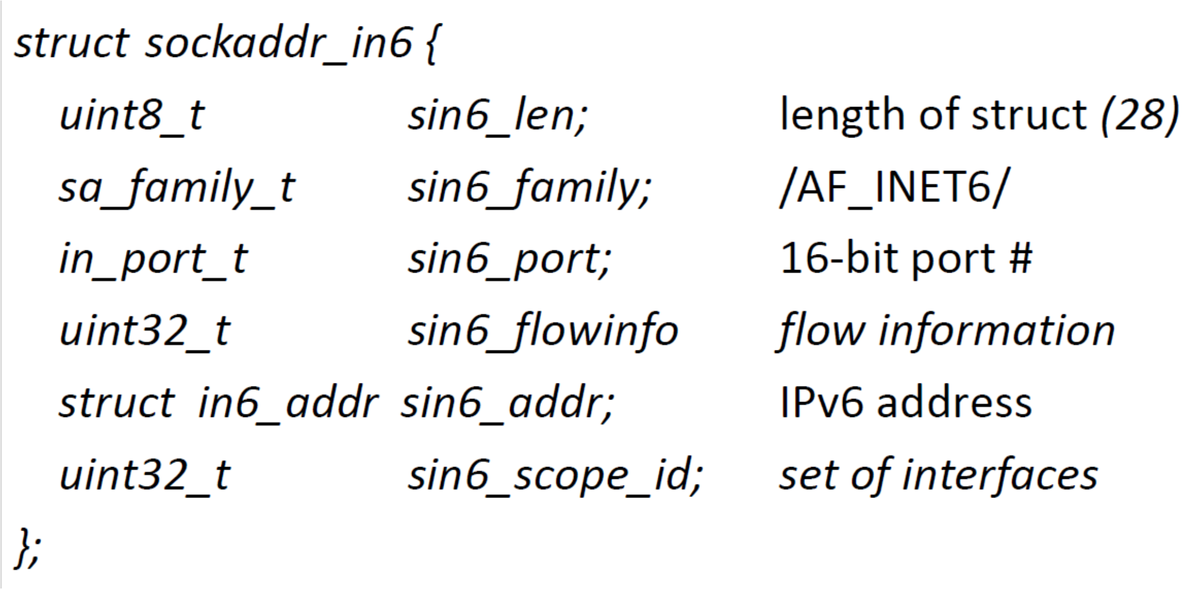
- **Security**

IPSec, which provides confidentiality, authentication and data integrity, is baked into in IPv6. Because of their potential to carry malware, IPv4 ICMP packets are often blocked by corporate firewalls, but ICMPv6, the implementation of the Internet Control Message Protocol for IPv6, may be permitted because IPSec can be applied to the ICMPv6 packets.

Difference between socket address structure of IPv4 and IPv6:

IPv6 socket address structure:

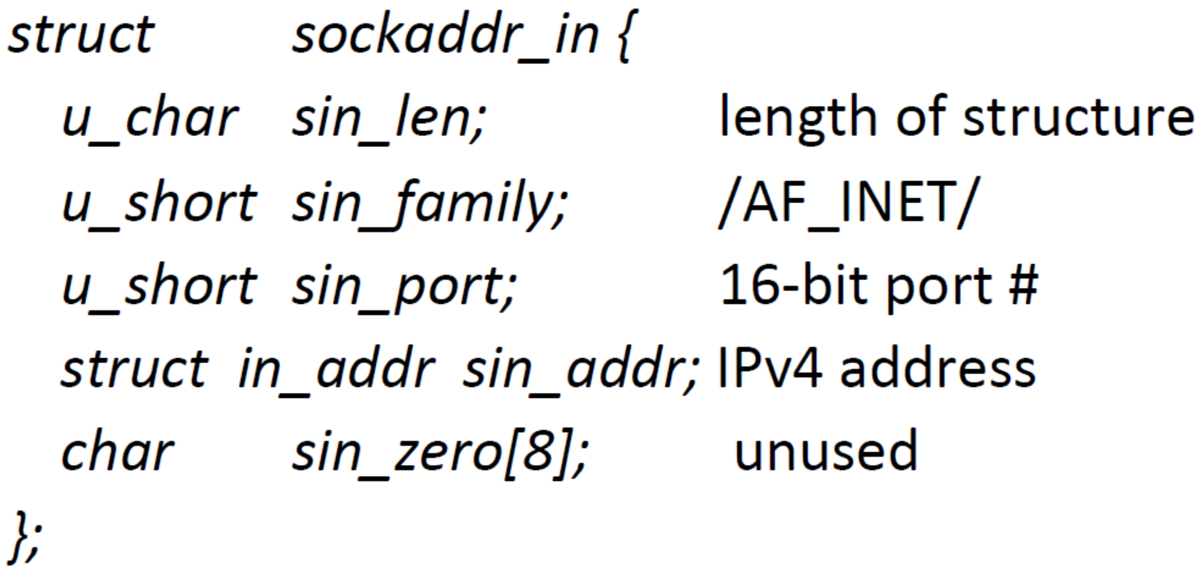
IPv6 socket address structure consists of extra two variables called sin6_flowinfo and sin6_scope_id as compared to IPv4 structure as shown below.



```
struct sockaddr_in6 {
   uint8_t           sin6_len;        length of struct (28)
   sa_family_t       sin6_family;     /AF_INET6/
   in_port_t         sin6_port;       16-bit port #
   uint32_t          sin6_flowinfo    flow information
   struct  in6_addr  sin6_addr;       IPv6 address
   uint32_t          sin6_scope_id;   set of interfaces
};
```

Fig2: IPv6 socket address structure

IPv4 socket address structure:



```
struct      sockaddr_in {
   u_char   sin_len;            length of structure
   u_short  sin_family;         /AF_INET/
   u_short  sin_port;           16-bit port #
   struct  in_addr  sin_addr;   IPv4 address
   char     sin_zero[8];         unused
};
```

Fig3: IPv4 socket address structure

Compatibility of IPv4 and IPv6:

IPv4/IPv6 co-existence can take one of three forms.

1. **Dual stack**, where your network hardware runs IPv4 and IPv6 simultaneously.
2. **Tunneling** one protocol within another. Usually, this means taking IPv6 packets and encapsulating them in IPv4 packets. The technical basics for these are outlined in the RFC 4213 Basic Transition Mechanisms for IPv6 Hosts and Routers.
3. **Network Address Translation-Protocol Translation** (NAT-PT) aka RFC-2766. This works just like the name says, software or a device translates IPv6 packets into IPv4 packets.

Our project has IPv6 server which is compatible with both of our IPv4 and IPv6 client.

2. I/O MULTIPLEXING

The capability to tell the kernel that we want to be notified if one or more I/O conditions are ready (i.e., input is ready to be read, or the descriptor is capable of taking more output). This capability is called I/O multiplexing and is provided by the **select** and **poll** functions.

I/O multiplexing is typically used in networking applications in the following scenarios:

1. When a client is handling multiple descriptors (normally interactive input and a network socket), I/O multiplexing should be used.
2. If a TCP server handles both a listening socket and its connected sockets, I/O multiplexing is normally used.
3. If a server handles both TCP and UDP, I/O multiplexing is normally used.
4. If a server handles multiple services and perhaps multiple protocols, I/O multiplexing is normally used.

Below are some of the I/O models used:

- Blocking I/O
- Non-Blocking I/O
- I/O multiplexing (select and poll)
- Signal driven I/O (SIGIO)
- Asynchronous I/O (the POSIX aio_functions)

I/O Multiplexing (select and poll)

With select(), instead of having a process for each request, there is usually only one process that "multi-plexes" all requests, servicing each request as much as it can.
So one main advantage of using select() is that your server will only require a single process to handle all requests. Thus, your server will not need shared memory or synchronization primitives for different 'tasks' to communicate.

We block in a call to select, waiting for the datagram socket to be readable. When select returns that the socket is readable, we then call recvfrom to copy the datagram into our application buffer

Below is the schematic shown for select call.

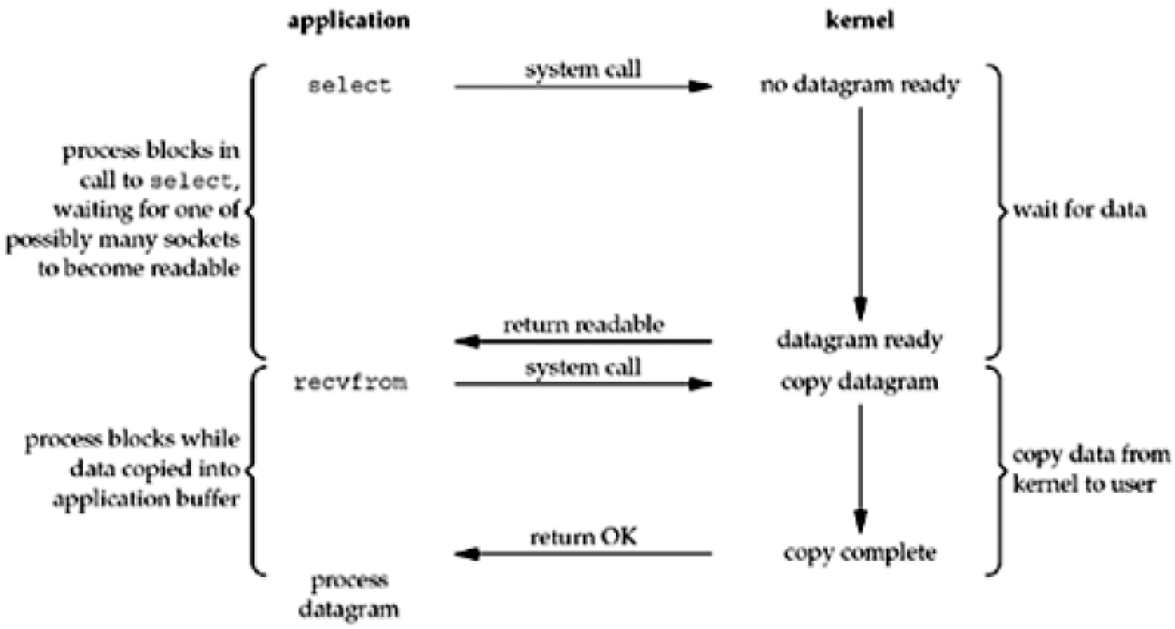

application
kernel
select
system call
no datagram ready
process blocks in call to select, waiting for one of possibly many sockets to become readable
wait for data
return readable
datagram ready
recvfrom
system call
copy datagram
process blocks while data copied into application buffer
copy data from kernel to user
return OK
copy complete
process datagram

Fig4: *Schematic of SELECT call*

Below is the syntax for select call:

int select(int nfds, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout)

nfds: # of file descriptors to be tested

readfds, **writefds**, **exceptfds**: file descriptors for reading, writing, exceptions

timeout: max time to wait; 0 no waiting

Select call returns number of ready file descriptors for success, 0 for time expire and -1 for error.

Few of the macros used along with select call are listed below:

- **fd_set (int fd, fd_set *fdset);**
  set the bit for fd in fdset.

- **fd_clr (int fd, fd_set *fdset);**
  clear the bit for fd in fdset.

- **fd_isset (int fd, fd_set *fdset);**
  test the bit for fd in fdset.

- **fd_zero (fd_set *fdset);**
  clear all bits in fdset.

3. SMTP: SIMPLE MAIL TRANSFER PROTOCOL

Simple Mail Transfer Protocol (SMTP) is an Internet standard for electronic mail (email) transmission. First defined by RFC 821 in 1982, it was last updated in 2008 with Extended SMTP additions by RFC 5321, which is the protocol in widespread use today.

Although electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically use SMTP only for sending messages to a mail server for relaying.

SMTP communication between mail servers uses port 25. Mail clients on the other hand, often submit the outgoing emails to a mail server on port 587 SMTP connections secured by SSL, known as SMTPS, default to port 465 (nonstandard, but sometimes used for legacy reasons).
SSL (Secure Sockets Layer) is the standard security technology for establishing an encrypted link between a web server and a browser. This link ensures that all data passed between the web server and browsers remain private and integral.

Most common SMTP Commands

- **EHLO** - introduce yourself and request extended mode
- **MAIL FROM**: - specify the sender
- **RCPT TO**: - specify the recipient
- **DATA** - specify the body of the message
- **RSET** - reset
- **QUIT** - quit the session
- **HELP** - get help on commands
- **VRFY** - verify an address
- **EXPN** - expand an address

FLOW DIAGRAM

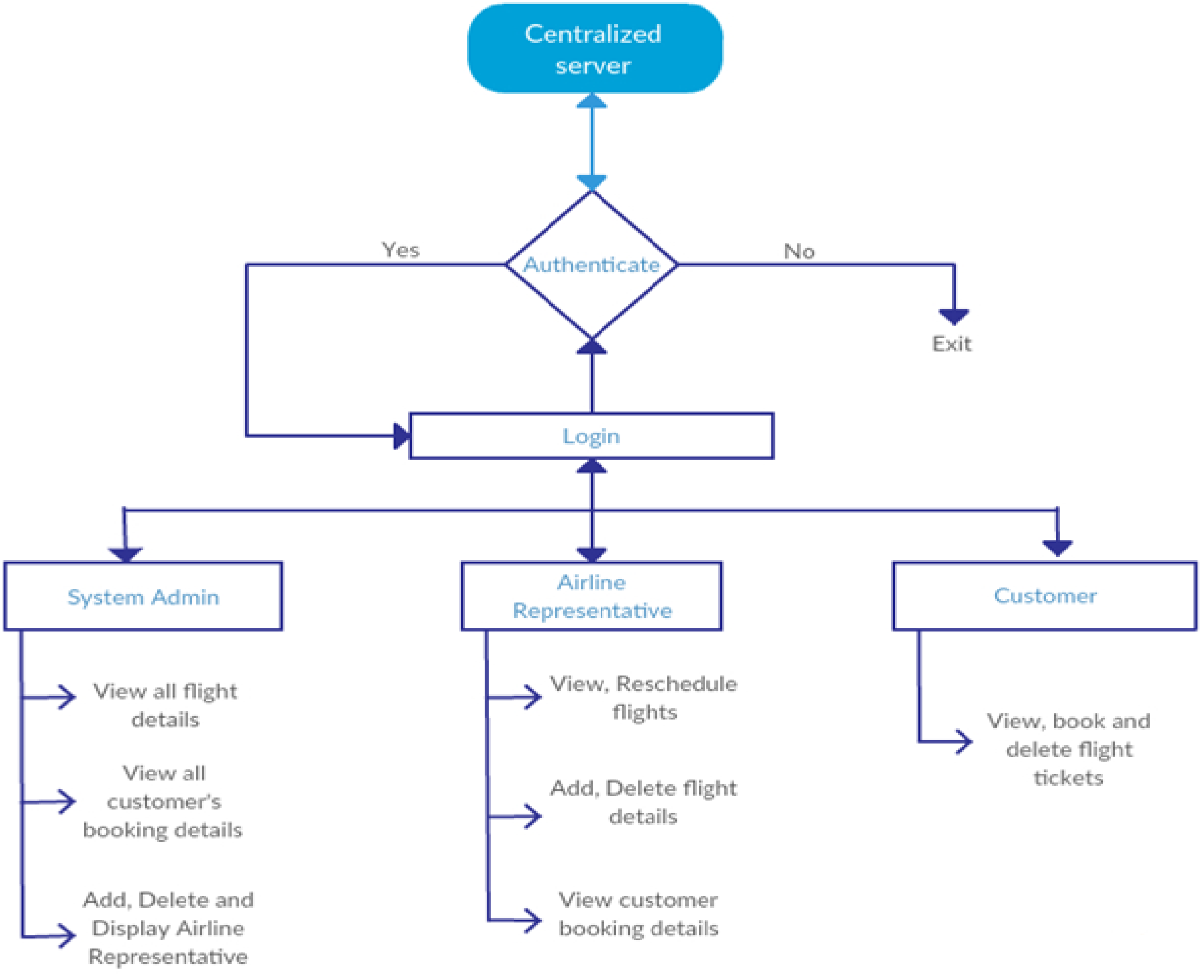

Centralized server
Yes
Authenticate
No
Exit
Login
System Admin
Airline Representative
Customer
View all flight details
View all customer's booking details
Add, Delete and Display Airline Representative
View, Reschedule flights
Add, Delete flight details
View customer booking details
View, book and delete flight tickets

Fig5: *Flow diagram*

Process flow

There is one centralized server and three players (clients) in the system as shown above.

Centralized Server:

Main objective of this server is to authenticate the users (clients). Users log in to the system using the username and password. Password is encrypted and sent along the network to the server which in turn searches the database for the matching username and password. If it exists a success message is sent else a failure message is sent to the client.

Clients:

System has 3 clients System admin, Airline representative and Customer. Each has their own set of functionalities in the system.

System Admin can view all flight details, view all customer booking details, add/delete/display airline representative details.

Airline representative can view/add/delete/reschedule their flight details. They can even view the customer booking details.

Customers can view/book/delete the flight tickets. Accordingly price and number of seats will be updated in database.

MODULES

Server

The main purpose of this centralized server is to authenticate the users (clients). The credentials entered by the users will be sent to server. The server looks for the entered credentials in the database, sends success or failure messages accordingly. If the success message is received at the client end, all the functionalities of player will be displayed on the terminal depending upon the roles. If failure message is sent by the server, client/users won't be able to do any functions.

Server concurrency is achieved through I/O multiplexing using select call. Select is a blocking call that waits for activity on file descriptors. Here, single process handle multiple connections. Since, the server function is only to authenticate the users, it requires little processing time to handle each client request.

Client

We have 3 clients in our application

1. System Administrator
2. Airline Representative
3. Customer

System Administrator will be added to the database at the time system boot up. He can only signs in into the system and perform below functions.

- Adding an Airline Representative
- View all Airline Representatives
- Deleting an Airline Representative
- View all Airline Representative's Flight Details
- View all Customer's Booking Details

Airline Representative can only log in as he will be added System Administrator. He can perform below functions.

- Adding a flight
- View his own flights
- Rescheduling flights
- Deleting flights
- View his Customer's Booking Details

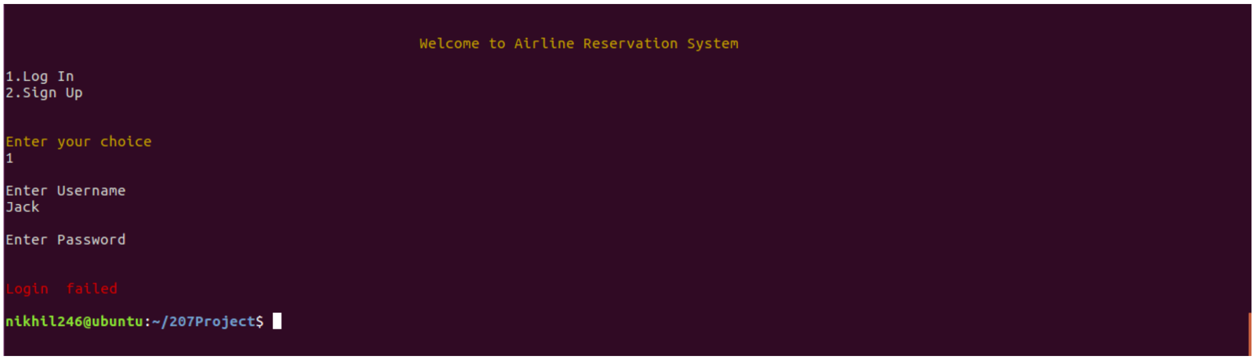
Customer has both login and sign up feature. New user has to sign up before he can use the application. He/she has enter the desired username on the registration form, if the username is already existing (used by another user) in the database, he will be prompted by suitable error message and asked to enter different username. Once user registers by giving unique username and password, user registration mail will be sent to the customer's mentioned email id. Customer can perform following functions.

- Book a flight ticket
- View Booking History
- Delete Booking

After the customer books a ticket, booking confirmation mail will be sent to mentioned email id. We have made use of SMTP (Simple Mail Transfer Protocol) connections secured by SSL (Secure Socket Layer) to send booking confirmation mail.

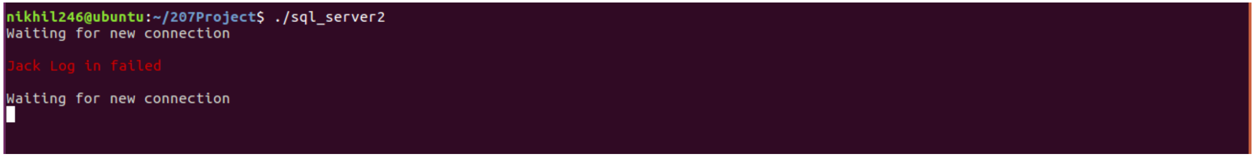
SCREENSHOTS

When un-registered user tries to login into system, server sends login failure message to client.



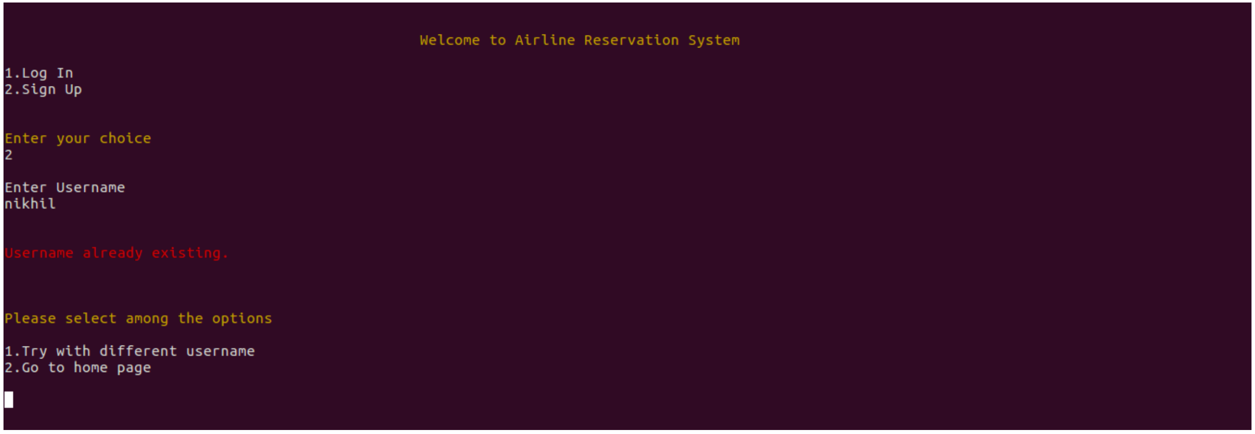
```
                                        Welcome to Airline Reservation System

1.Log In
2.Sign Up

Enter your choice
1

Enter Username
Jack

Enter Password


Login  failed

nikhil246@ubuntu:~/207Project$ 
```

Error message displayed at server side when an un-registered user tries to login



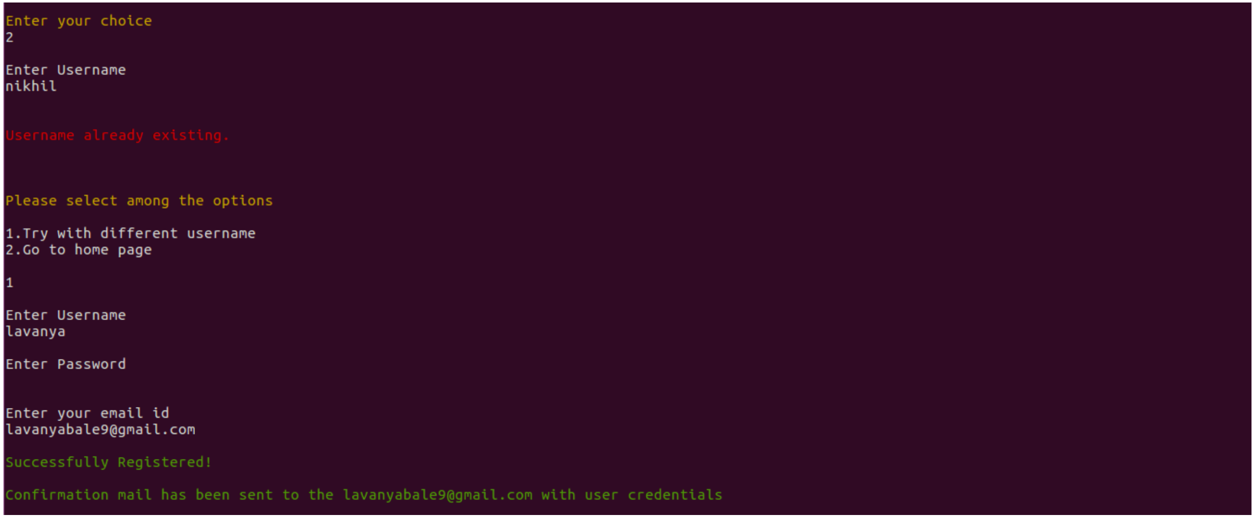
```
nikhil246@ubuntu:~/207Project$ ./sql_server2
Waiting for new connection

Jack Log in failed

Waiting for new connection
```

When a new user tries to register with the username that is already existing in the database, proper error message will be displayed.



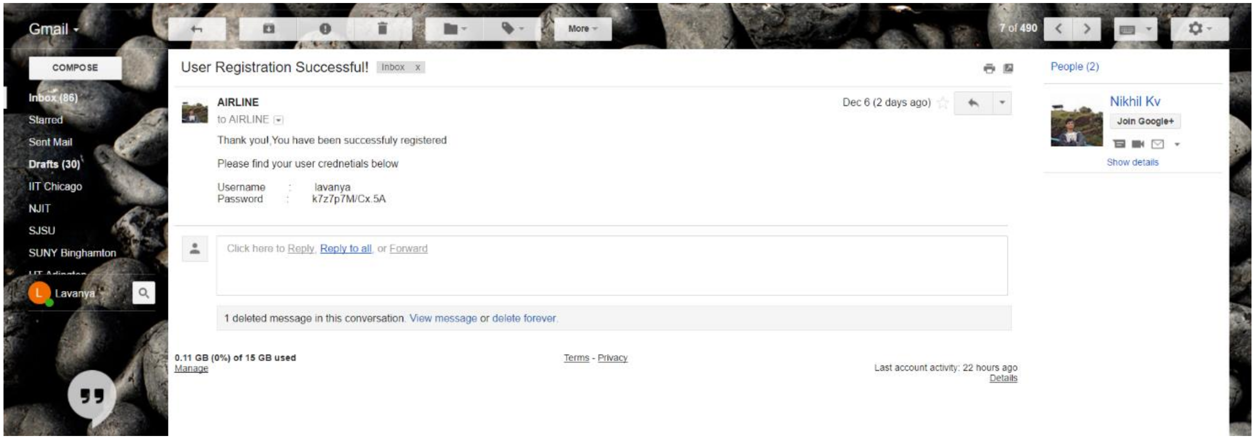
```
                                        Welcome to Airline Reservation System

1.Log In
2.Sign Up

Enter your choice
2

Enter Username
nikhil


Username already existing.


Please select among the options

1.Try with different username
2.Go to home page
```

Registration page for new user – Sign Up feature



```
Enter your choice
2

Enter Username
nikhil


Username already existing.



Please select among the options

1.Try with different username
2.Go to home page

1

Enter Username
lavanya

Enter Password


Enter your email id
lavanyabale9@gmail.com

Successfully Registered!

Confirmation mail has been sent to the lavanyabale9@gmail.com with user credentials
```

User Registration mail will be sent to new registered user with login credentials.



User Registration Successful! Inbox x

AIRLINE
to AIRLINE

Dec 6 (2 days ago)

Thank you! You have been successfuly registered

Please find your user crednetials below

Username : lavanya
Password : k7z7p7M/Cx.5A

Click here to Reply, Reply to all, or Forward

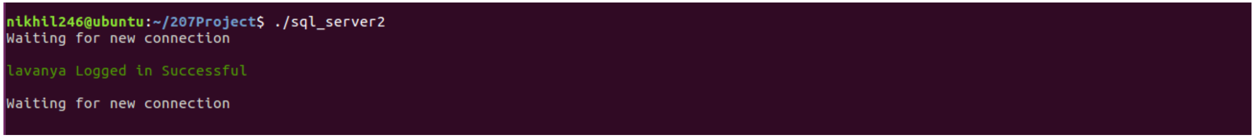
1 deleted message in this conversation. View message or delete forever.

0.11 GB (0%) of 15 GB used
Manage

Terms - Privacy

Last account activity: 22 hours ago
Details

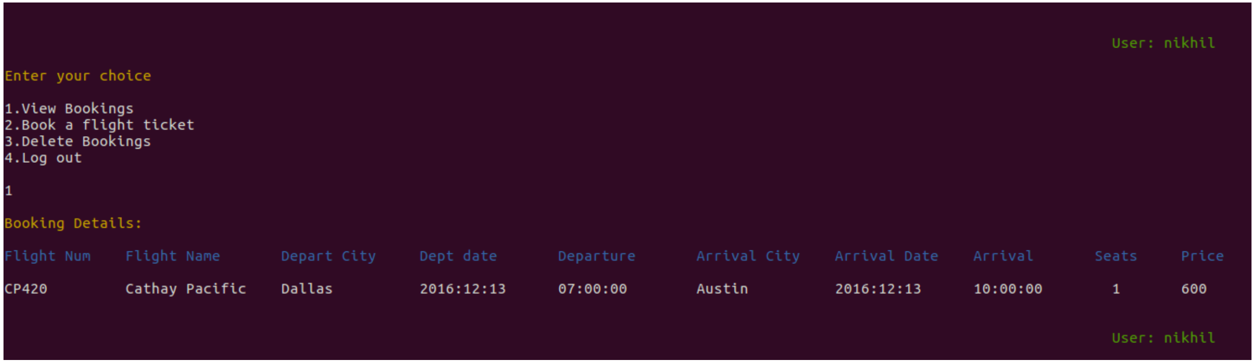
Server authenticates registered user with success message.



```
nikhil246@ubuntu:~/207Project$ ./sql_server2
Waiting for new connection

lavanya Logged in Successful

Waiting for new connection
```

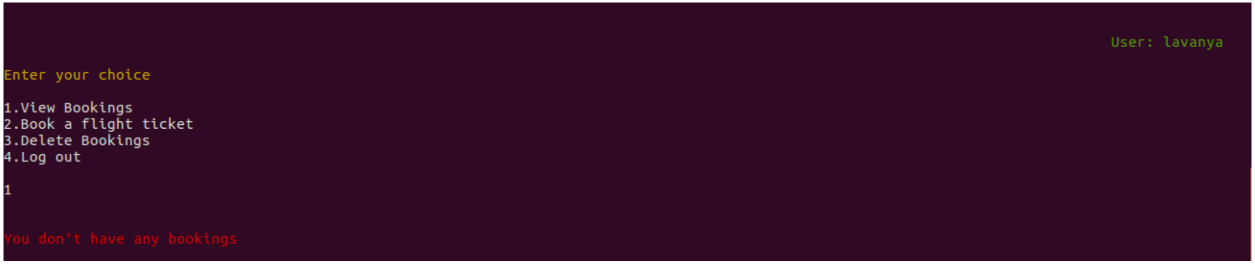
Functionalities offered to Customer:

Customer viewing his booking details



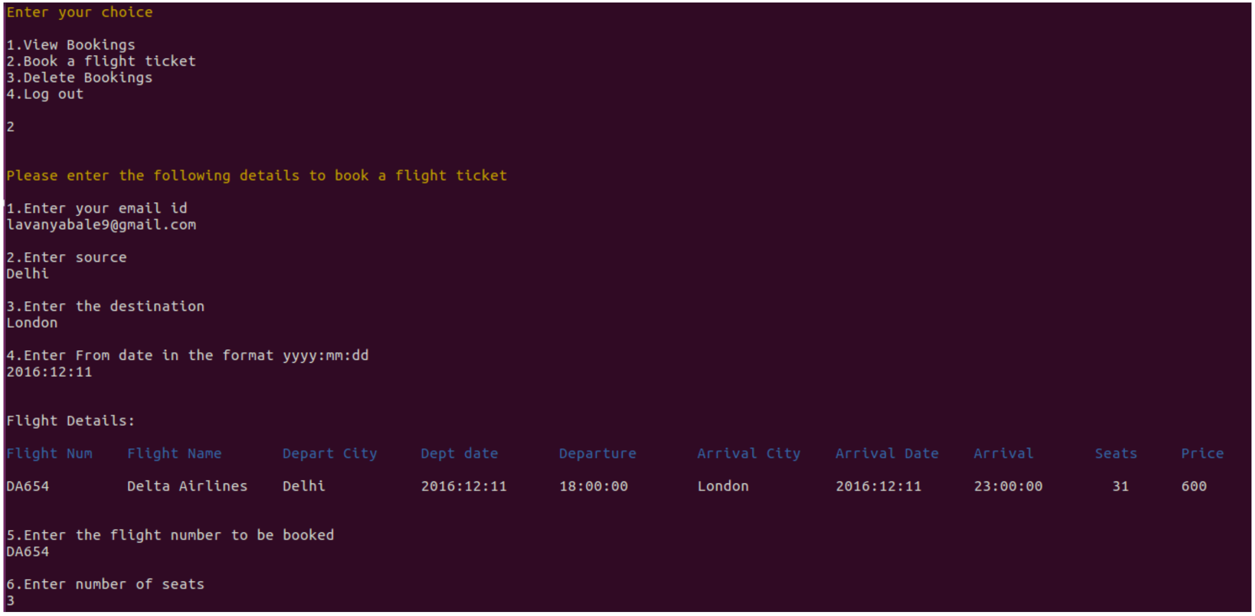
```
                                                                                                                                                          User: nikhil

Enter your choice

1.View Bookings
2.Book a flight ticket
3.Delete Bookings
4.Log out

1

Booking Details:

Flight Num    Flight Name       Depart City      Dept date        Departure        Arrival City     Arrival Date     Arrival       Seats     Price

CP420         Cathay Pacific    Dallas           2016:12:13       07:00:00         Austin           2016:12:13       10:00:00       1        600

                                                                                                                                                          User: nikhil
```

If a newly registered user tries to view his bookings, he will prompted with suitable message. Proper error handling is done.



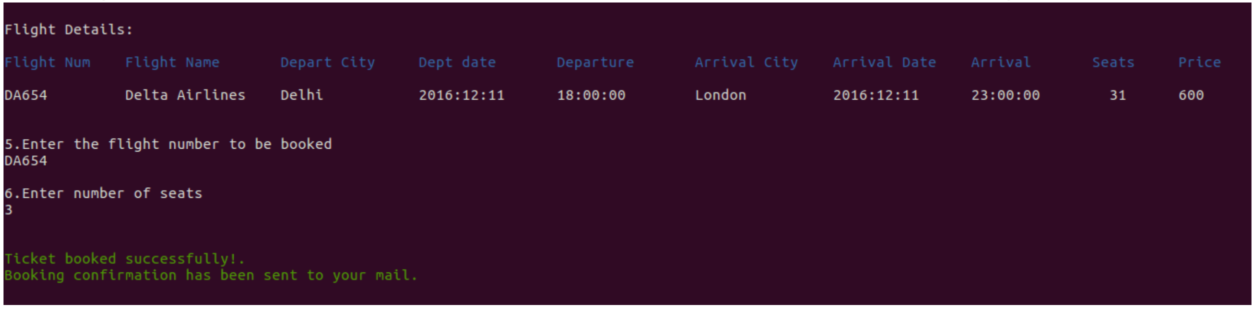
```
                                                                                                                                                          User: lavanya

Enter your choice

1.View Bookings
2.Book a flight ticket
3.Delete Bookings
4.Log out

1


You don't have any bookings
```

Customer ticket booking form



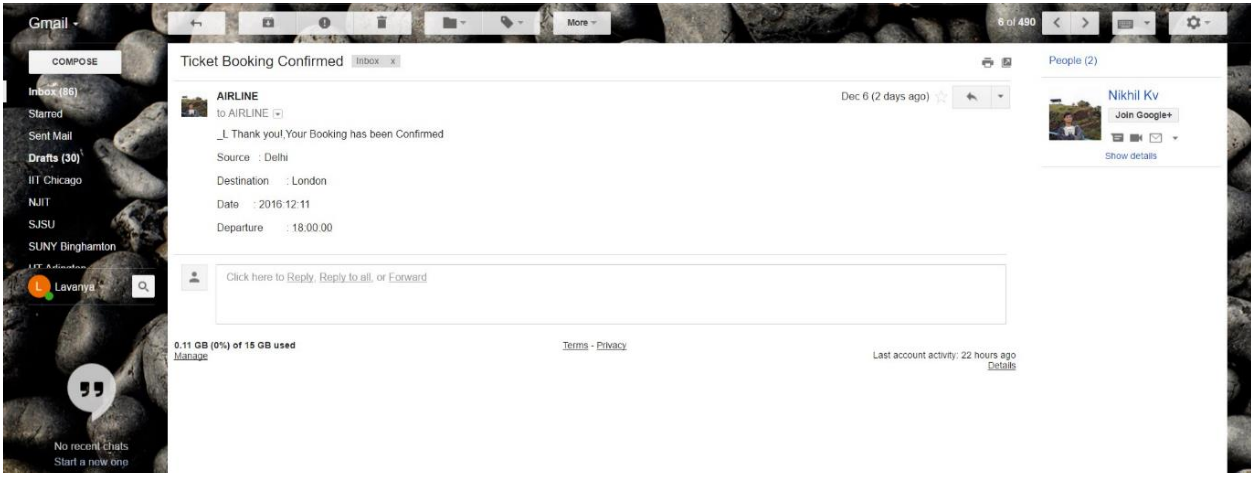
```
Enter your choice

1.View Bookings
2.Book a flight ticket
3.Delete Bookings
4.Log out

2


Please enter the following details to book a flight ticket

1.Enter your email id
lavanyabale9@gmail.com

2.Enter source
Delhi

3.Enter the destination
London

4.Enter From date in the format yyyy:mm:dd
2016:12:11


Flight Details:

Flight Num    Flight Name       Depart City      Dept date        Departure        Arrival City     Arrival Date     Arrival       Seats     Price

DA654         Delta Airlines    Delhi            2016:12:11       18:00:00         London           2016:12:11       23:00:00       31       600


5.Enter the flight number to be booked
DA654

6.Enter number of seats
3
```

Booking confirmation details will be sent to customer email-id after booking is done.



```
Flight Details:

Flight Num    Flight Name      Depart City     Dept date       Departure       Arrival City    Arrival Date    Arrival       Seats     Price

DA654         Delta Airlines   Delhi           2016:12:11      18:00:00        London          2016:12:11      23:00:00        31      600


5.Enter the flight number to be booked
DA654

6.Enter number of seats
3


Ticket booked successfully!.
Booking confirmation has been sent to your mail.
```

Booking Confirmation mail sent to customer



Ticket Booking Confirmed

AIRLINE
to AIRLINE
Dec 6 (2 days ago)

_L Thank you!,Your Booking has been Confirmed

Source : Delhi

Destination : London

Date : 2016:12:11

Departure : 18:00:00

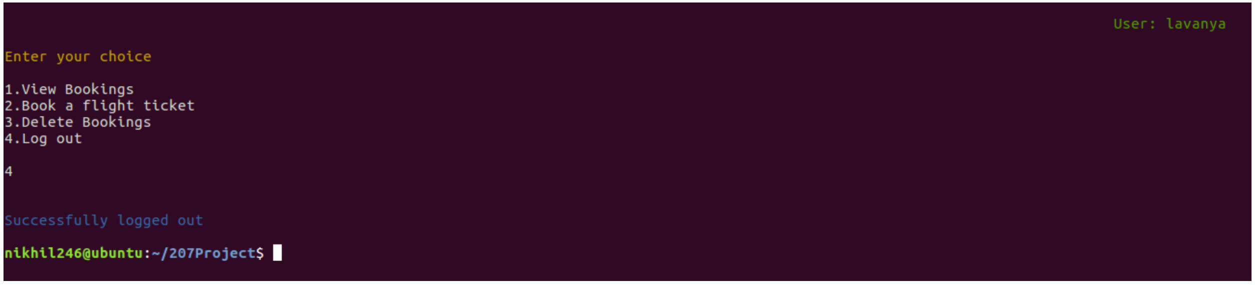
Click here to Reply, Reply to all, or Forward

0.11 GB (0%) of 15 GB used
Manage

Terms - Privacy

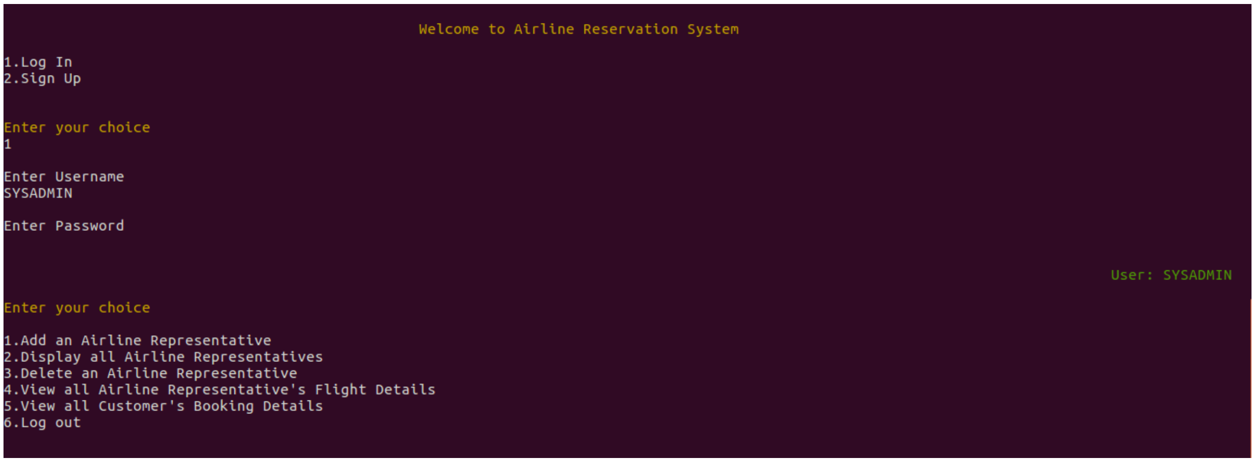
Last account activity: 22 hours ago
Details

Log out feature for customer



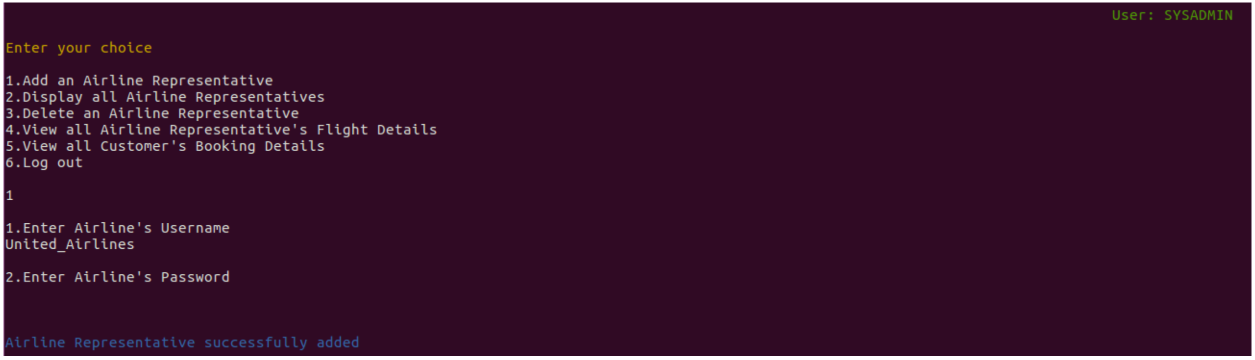
```
                                                                                                                            User: lavanya

Enter your choice

1.View Bookings
2.Book a flight ticket
3.Delete Bookings
4.Log out

4


Successfully logged out

nikhil246@ubuntu:~/207Project$ 
```

Functionalities offered to System Administrator



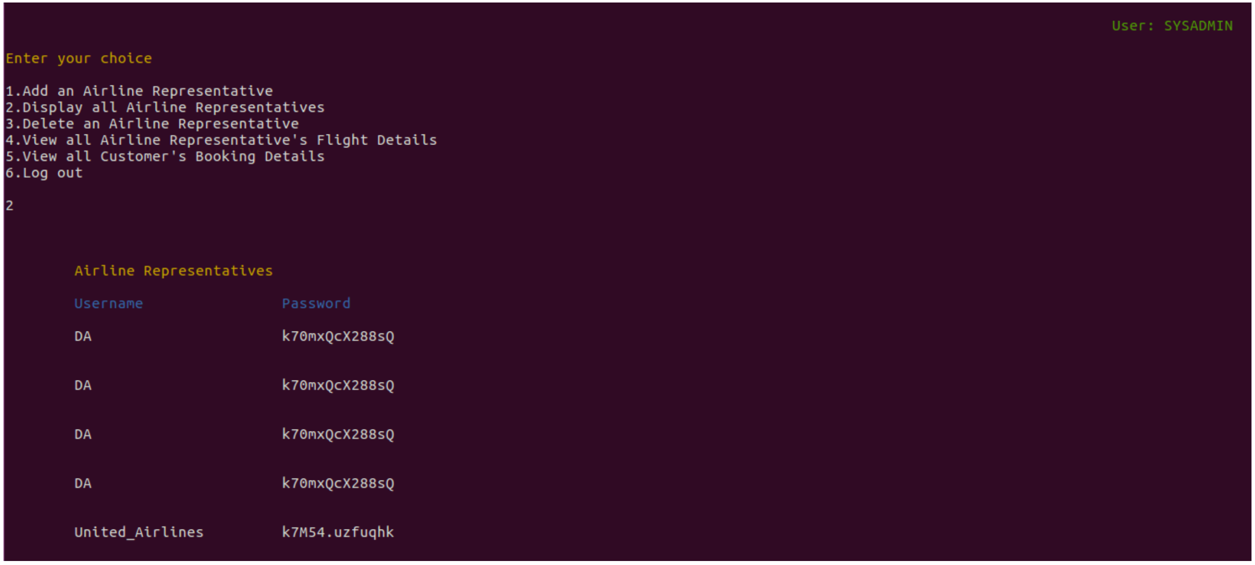
```
                                           Welcome to Airline Reservation System

1.Log In
2.Sign Up


Enter your choice
1

Enter Username
SYSADMIN

Enter Password


                                                                                                        User: SYSADMIN

Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out
```

Adding an Airline Representative



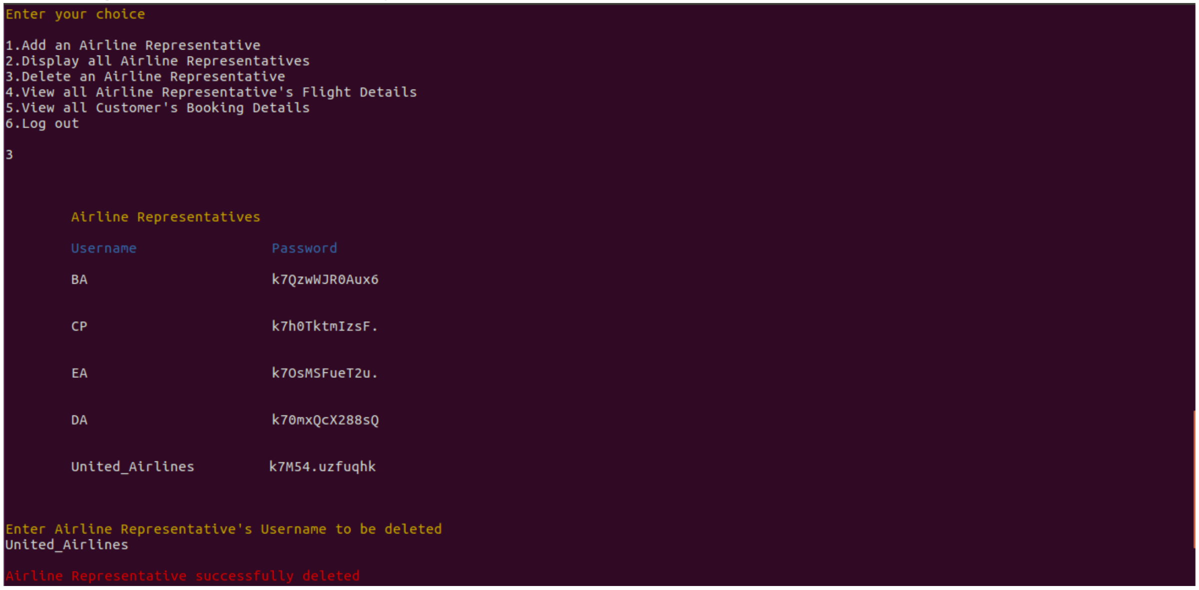
```
                                                                                                        User: SYSADMIN

Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out

1

1.Enter Airline's Username
United_Airlines

2.Enter Airline's Password



Airline Representative successfully added
```

Displaying all Airline Representatives



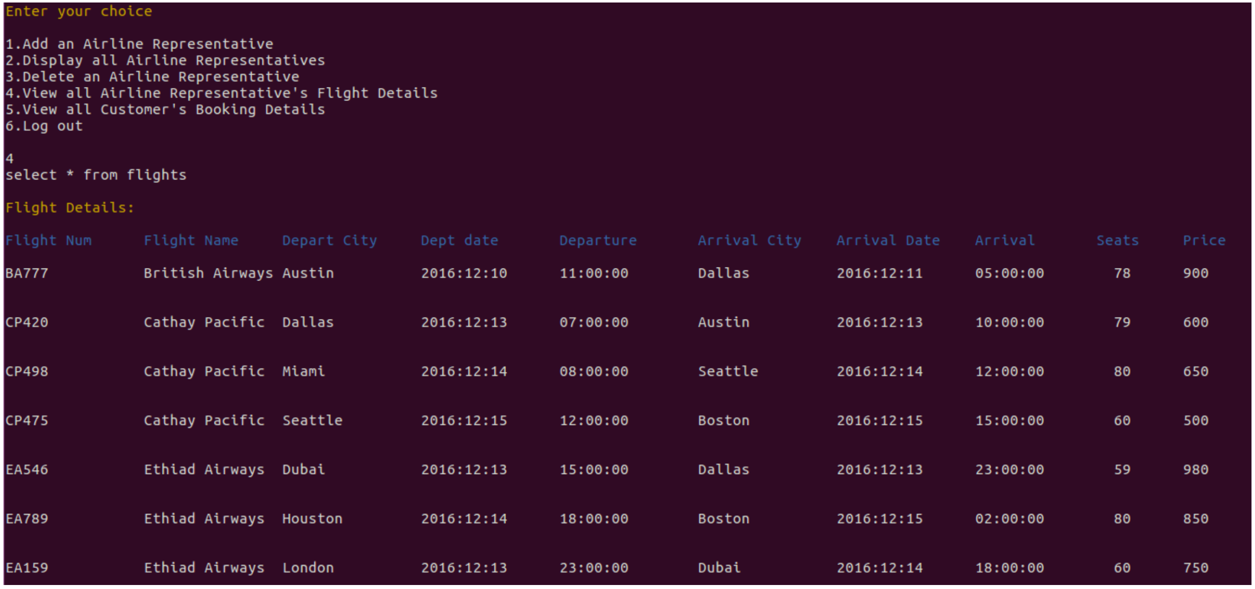
```
                                                                                                        User: SYSADMIN

Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out

2



        Airline Representatives

        Username                Password

        DA                      k70mxQcX288sQ


        DA                      k70mxQcX288sQ


        DA                      k70mxQcX288sQ


        DA                      k70mxQcX288sQ


        United_Airlines         k7M54.uzfuqhk
```

Deleting an added Airline Representative



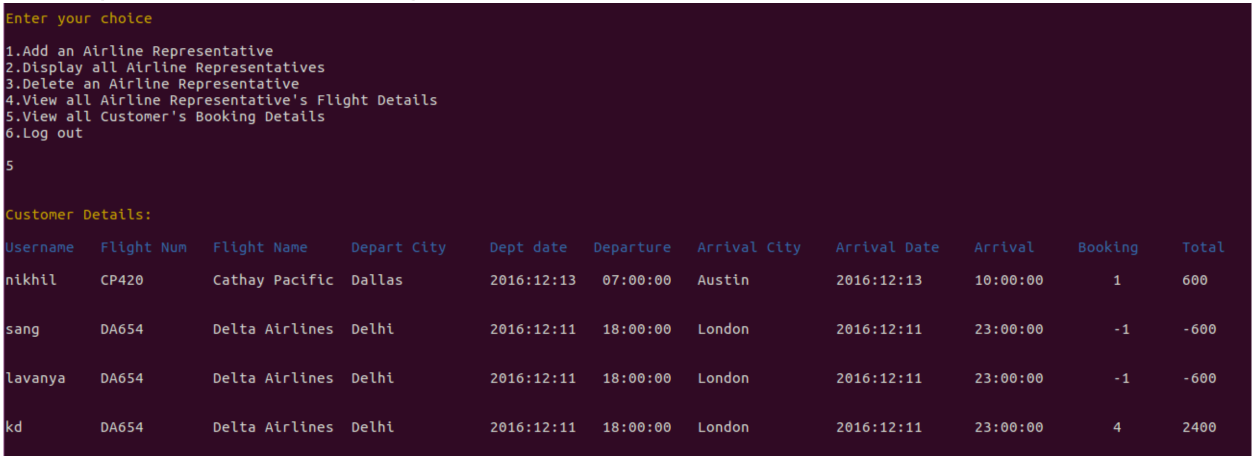
```
Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out

3



        Airline Representatives

        Username                 Password

        BA                       k7QzwWJR0Aux6


        CP                       k7h0TktmIzsF.


        EA                       k7OsMSFueT2u.


        DA                       k70mxQcX288sQ


        United_Airlines         k7M54.uzfuqhk


Enter Airline Representative's Username to be deleted
United_Airlines

Airline Representative successfully deleted
```

Viewing all Airline Representative’s flight details



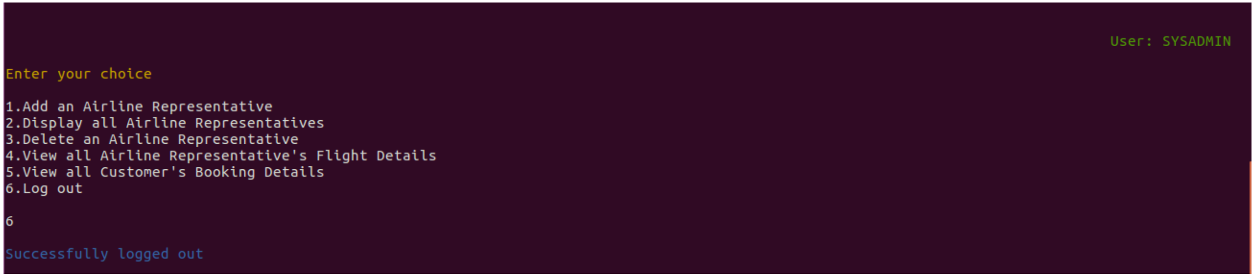
```
Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out

4
select * from flights

Flight Details:

Flight Num       Flight Name      Depart City      Dept date        Departure        Arrival City     Arrival Date     Arrival          Seats      Price

BA777            British Airways  Austin           2016:12:10       11:00:00         Dallas           2016:12:11       05:00:00          78        900


CP420            Cathay Pacific   Dallas           2016:12:13       07:00:00         Austin           2016:12:13       10:00:00          79        600


CP498            Cathay Pacific   Miami            2016:12:14       08:00:00         Seattle          2016:12:14       12:00:00          80        650


CP475            Cathay Pacific   Seattle          2016:12:15       12:00:00         Boston           2016:12:15       15:00:00          60        500


EA546            Ethiad Airways   Dubai            2016:12:13       15:00:00         Dallas           2016:12:13       23:00:00          59        980


EA789            Ethiad Airways   Houston          2016:12:14       18:00:00         Boston           2016:12:15       02:00:00          80        850


EA159            Ethiad Airways   London           2016:12:13       23:00:00         Dubai            2016:12:14       18:00:00          60        750
```

Viewing all Customer's Booking Details



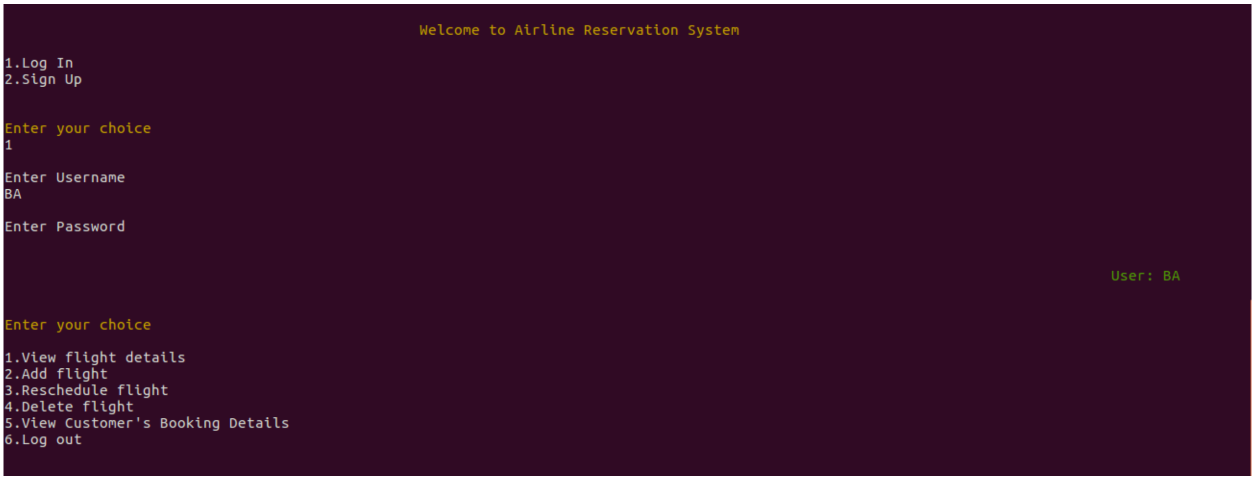
```
Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out

5


Customer Details:

Username    Flight Num    Flight Name     Depart City     Dept date    Departure    Arrival City     Arrival Date    Arrival     Booking     Total

nikhil      CP420         Cathay Pacific  Dallas          2016:12:13    07:00:00    Austin           2016:12:13      10:00:00       1        600


sang        DA654         Delta Airlines  Delhi           2016:12:11    18:00:00    London           2016:12:11      23:00:00       -1       -600


lavanya     DA654         Delta Airlines  Delhi           2016:12:11    18:00:00    London           2016:12:11      23:00:00       -1       -600


kd          DA654         Delta Airlines  Delhi           2016:12:11    18:00:00    London           2016:12:11      23:00:00       4        2400
```

Logout feature for SYSADMIN



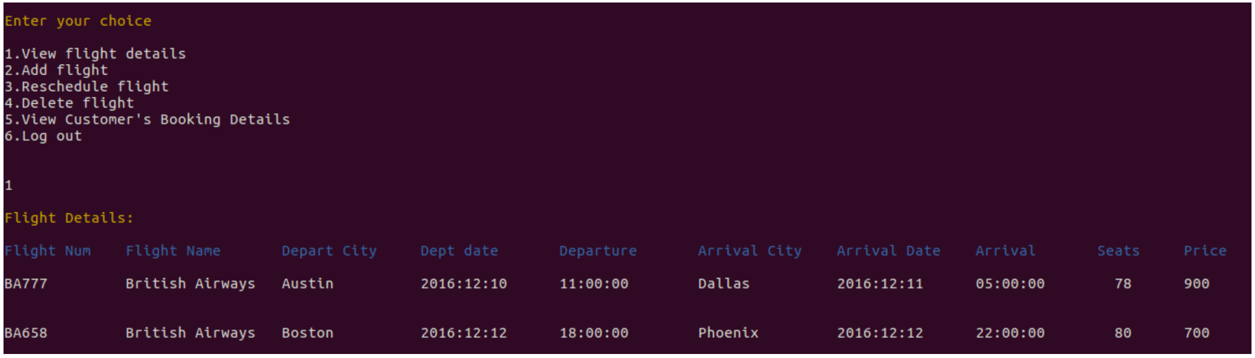
```
                                                                                                                                    User: SYSADMIN

Enter your choice

1.Add an Airline Representative
2.Display all Airline Representatives
3.Delete an Airline Representative
4.View all Airline Representative's Flight Details
5.View all Customer's Booking Details
6.Log out

6

Successfully logged out
```

Functionalities offered to Airline Representative



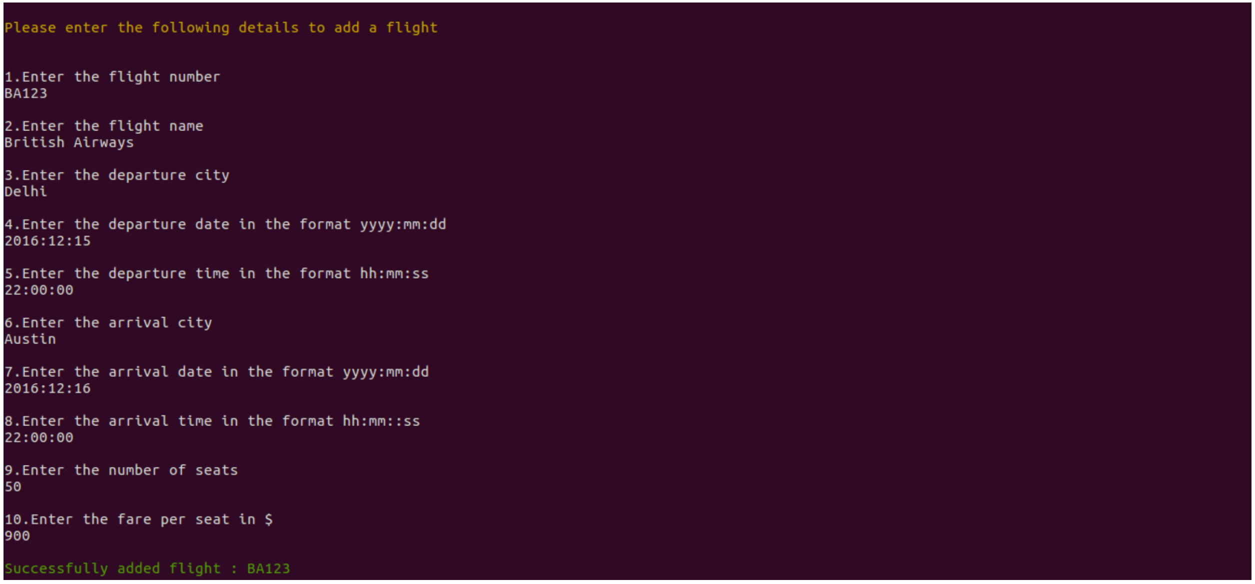
```
                                               Welcome to Airline Reservation System

1.Log In
2.Sign Up


Enter your choice
1

Enter Username
BA

Enter Password


                                                                                                                                       User: BA


Enter your choice

1.View flight details
2.Add flight
3.Reschedule flight
4.Delete flight
5.View Customer's Booking Details
6.Log out
```

Viewing added flights



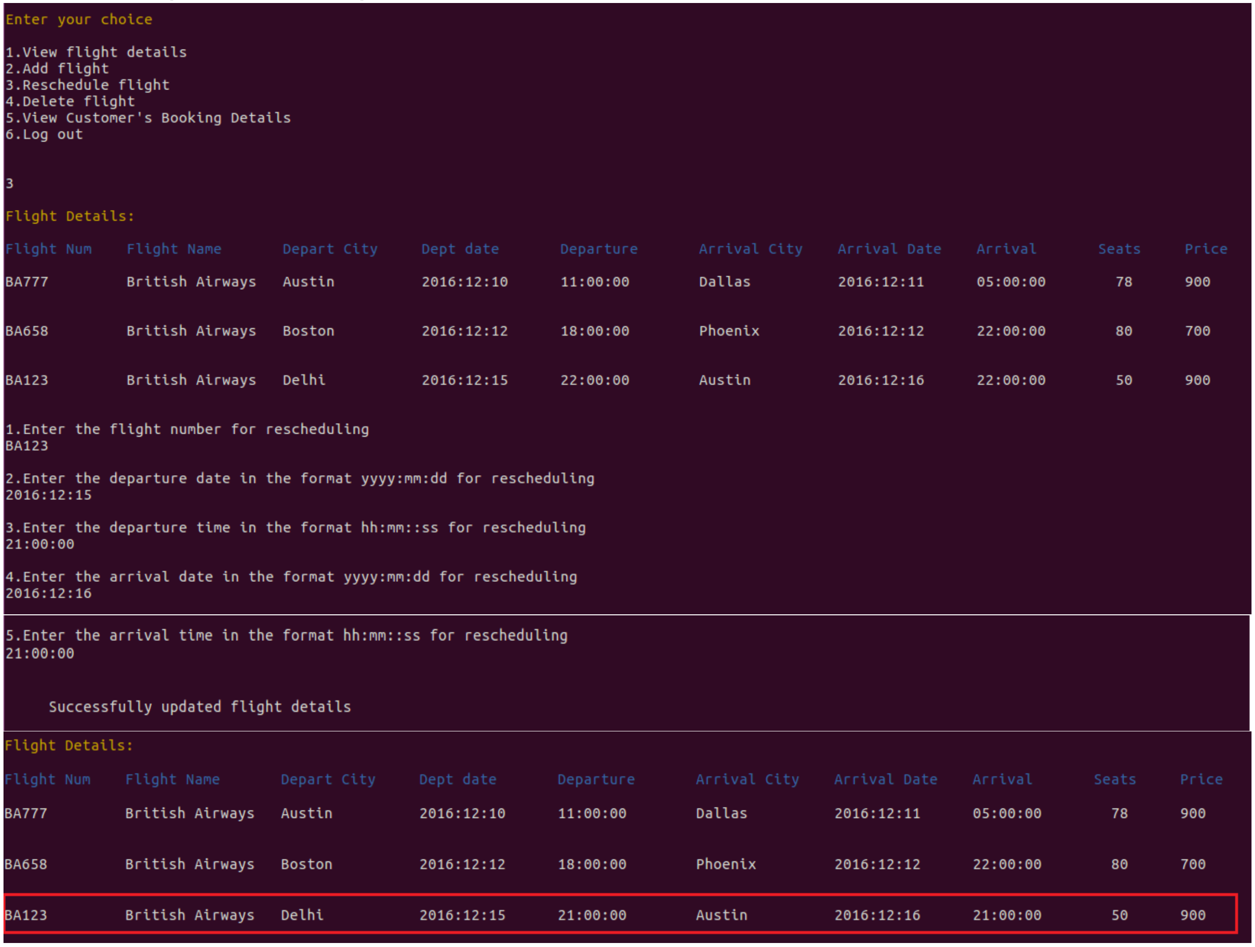
```
Enter your choice

1.View flight details
2.Add flight
3.Reschedule flight
4.Delete flight
5.View Customer's Booking Details
6.Log out


1

Flight Details:

Flight Num    Flight Name      Depart City      Dept date        Departure        Arrival City     Arrival Date     Arrival       Seats     Price

BA777         British Airways  Austin           2016:12:10       11:00:00         Dallas           2016:12:11       05:00:00       78       900


BA658         British Airways  Boston           2016:12:12       18:00:00         Phoenix          2016:12:12       22:00:00       80       700
```

Adding a new flight



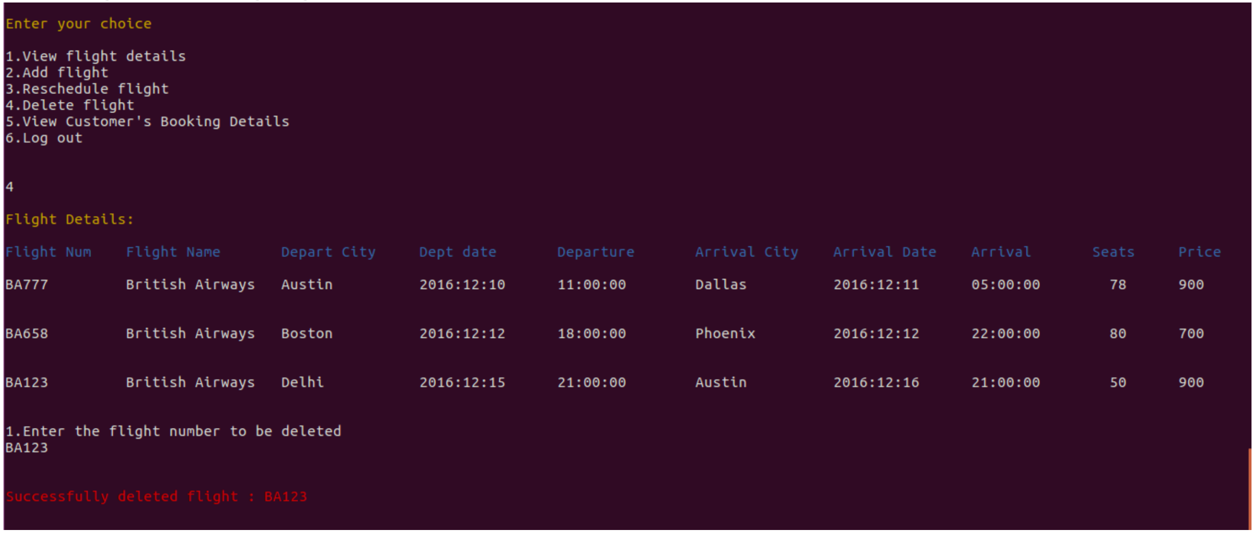
```
Please enter the following details to add a flight


1.Enter the flight number
BA123

2.Enter the flight name
British Airways

3.Enter the departure city
Delhi

4.Enter the departure date in the format yyyy:mm:dd
2016:12:15

5.Enter the departure time in the format hh:mm:ss
22:00:00

6.Enter the arrival city
Austin

7.Enter the arrival date in the format yyyy:mm:dd
2016:12:16

8.Enter the arrival time in the format hh:mm::ss
22:00:00

9.Enter the number of seats
50

10.Enter the fare per seat in $
900

Successfully added flight : BA123
```

Rescheduling the added flight



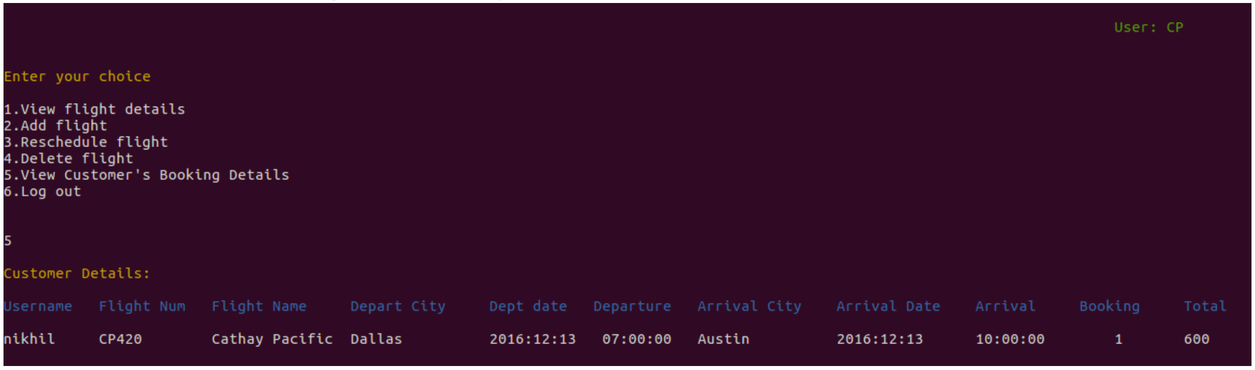
```
Enter your choice

1.View flight details
2.Add flight
3.Reschedule flight
4.Delete flight
5.View Customer's Booking Details
6.Log out


3

Flight Details:

Flight Num    Flight Name       Depart City     Dept date       Departure       Arrival City    Arrival Date    Arrival       Seats     Price

BA777         British Airways   Austin          2016:12:10      11:00:00        Dallas          2016:12:11      05:00:00        78      900


BA658         British Airways   Boston          2016:12:12      18:00:00        Phoenix         2016:12:12      22:00:00        80      700


BA123         British Airways   Delhi           2016:12:15      22:00:00        Austin          2016:12:16      22:00:00        50      900


1.Enter the flight number for rescheduling
BA123

2.Enter the departure date in the format yyyy:mm:dd for rescheduling
2016:12:15

3.Enter the departure time in the format hh:mm::ss for rescheduling
21:00:00

4.Enter the arrival date in the format yyyy:mm:dd for rescheduling
2016:12:16

5.Enter the arrival time in the format hh:mm::ss for rescheduling
21:00:00


     Successfully updated flight details

Flight Details:

Flight Num    Flight Name       Depart City     Dept date       Departure       Arrival City    Arrival Date    Arrival       Seats     Price

BA777         British Airways   Austin          2016:12:10      11:00:00        Dallas          2016:12:11      05:00:00        78      900


BA658         British Airways   Boston          2016:12:12      18:00:00        Phoenix         2016:12:12      22:00:00        80      700


BA123         British Airways   Delhi           2016:12:15      21:00:00        Austin          2016:12:16      21:00:00        50      900
```

Deleting an existing flight



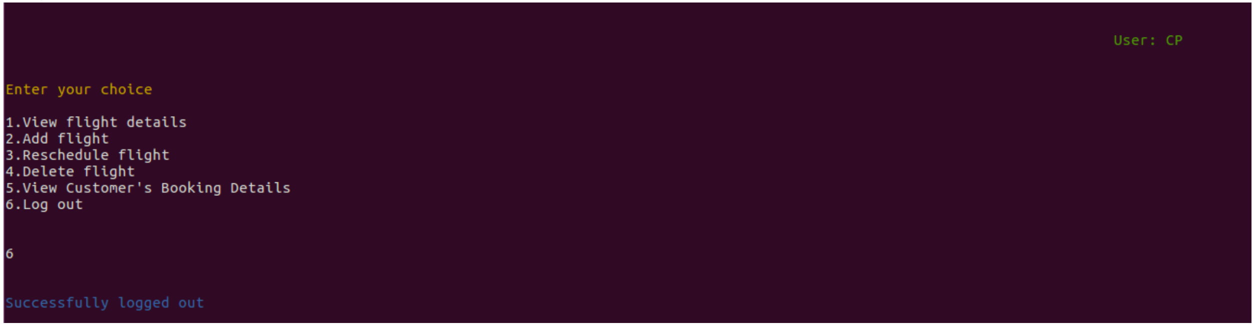
```
Enter your choice

1.View flight details
2.Add flight
3.Reschedule flight
4.Delete flight
5.View Customer's Booking Details
6.Log out


4

Flight Details:

Flight Num    Flight Name       Depart City     Dept date       Departure       Arrival City    Arrival Date    Arrival       Seats     Price

BA777         British Airways   Austin          2016:12:10      11:00:00        Dallas          2016:12:11      05:00:00        78      900


BA658         British Airways   Boston          2016:12:12      18:00:00        Phoenix         2016:12:12      22:00:00        80      700


BA123         British Airways   Delhi           2016:12:15      21:00:00        Austin          2016:12:16      21:00:00        50      900


1.Enter the flight number to be deleted
BA123


Successfully deleted flight : BA123
```

View Customer's Booking Details of specific Airline



```
                                                                                                                              User: CP


Enter your choice

1.View flight details
2.Add flight
3.Reschedule flight
4.Delete flight
5.View Customer's Booking Details
6.Log out


5

Customer Details:

Username   Flight Num   Flight Name     Depart City      Dept date    Departure   Arrival City     Arrival Date     Arrival     Booking     Total

nikhil     CP420        Cathay Pacific  Dallas           2016:12:13    07:00:00   Austin           2016:12:13       10:00:00       1        600
```

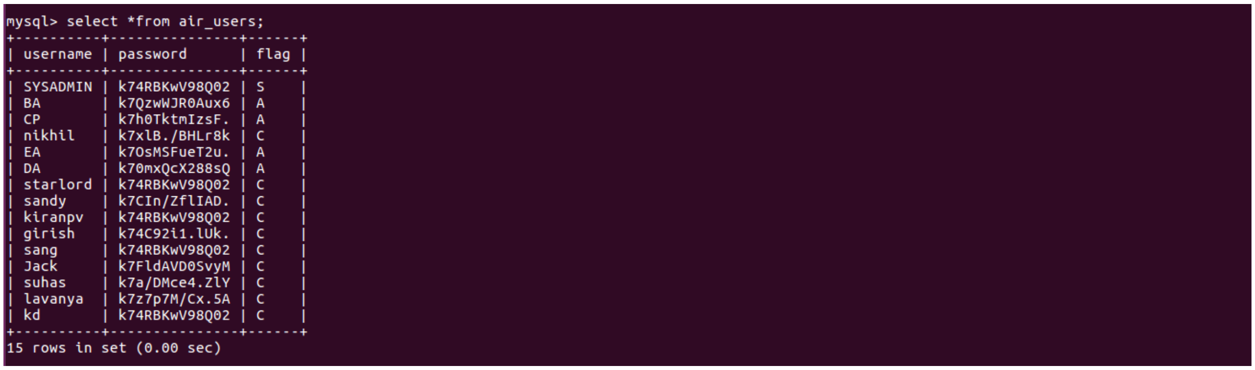
Log out feature for Airline Representative



```
                                                                                                                              User: CP


Enter your choice

1.View flight details
2.Add flight
3.Reschedule flight
4.Delete flight
5.View Customer's Booking Details
6.Log out


6


Successfully logged out
```

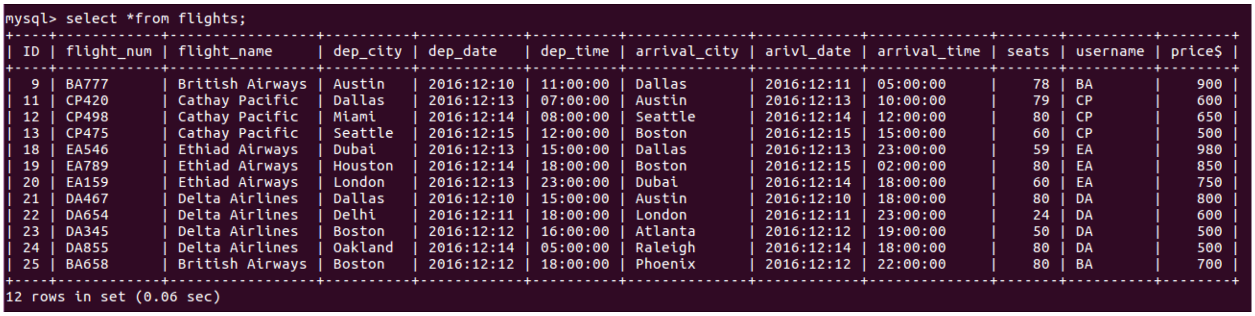
Tables Used:

1. air_users



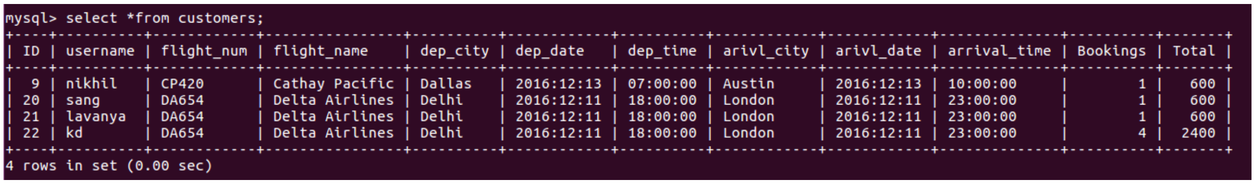
```
mysql> select *from air_users;
+----------+---------------+------+
| username | password      | flag |
+----------+---------------+------+
| SYSADMIN | k74RBKwV98Q02 | S    |
| BA       | k7QzwWJR0Aux6 | A    |
| CP       | k7h0TktmIzsF. | A    |
| nikhil   | k7xlB./BHLr8k | C    |
| EA       | k7OsMSFueT2u. | A    |
| DA       | k70mxQcX288sQ | A    |
| starlord | k74RBKwV98Q02 | C    |
| sandy    | k7CIn/ZflIAD. | C    |
| kiranpv  | k74RBKwV98Q02 | C    |
| girish   | k74C92i1.lUk. | C    |
| sang     | k74RBKwV98Q02 | C    |
| Jack     | k7FldAVD0SvyM | C    |
| suhas    | k7a/DMce4.ZlY | C    |
| lavanya  | k7z7p7M/Cx.5A | C    |
| kd       | k74RBKwV98Q02 | C    |
+----------+---------------+------+
15 rows in set (0.00 sec)
```

2. flights



```
mysql> select *from flights;
+----+------------+-----------------+----------+------------+----------+--------------+------------+--------------+-------+----------+--------+
| ID | flight_num | flight_name     | dep_city | dep_date   | dep_time | arrival_city | arivl_date | arrival_time | seats | username | price$ |
+----+------------+-----------------+----------+------------+----------+--------------+------------+--------------+-------+----------+--------+
|  9 | BA777      | British Airways | Austin   | 2016:12:10 | 11:00:00 | Dallas       | 2016:12:11 | 05:00:00     |    78 | BA       |    900 |
| 11 | CP420      | Cathay Pacific  | Dallas   | 2016:12:13 | 07:00:00 | Austin       | 2016:12:13 | 10:00:00     |    79 | CP       |    600 |
| 12 | CP498      | Cathay Pacific  | Miami    | 2016:12:14 | 08:00:00 | Seattle      | 2016:12:14 | 12:00:00     |    80 | CP       |    650 |
| 13 | CP475      | Cathay Pacific  | Seattle  | 2016:12:15 | 12:00:00 | Boston       | 2016:12:15 | 15:00:00     |    60 | CP       |    500 |
| 18 | EA546      | Ethiad Airways  | Dubai    | 2016:12:13 | 15:00:00 | Dallas       | 2016:12:13 | 23:00:00     |    59 | EA       |    980 |
| 19 | EA789      | Ethiad Airways  | Houston  | 2016:12:14 | 18:00:00 | Boston       | 2016:12:15 | 02:00:00     |    80 | EA       |    850 |
| 20 | EA159      | Ethiad Airways  | London   | 2016:12:13 | 23:00:00 | Dubai        | 2016:12:14 | 18:00:00     |    60 | EA       |    750 |
| 21 | DA467      | Delta Airlines  | Dallas   | 2016:12:10 | 15:00:00 | Austin       | 2016:12:10 | 18:00:00     |    80 | DA       |    800 |
| 22 | DA654      | Delta Airlines  | Delhi    | 2016:12:11 | 18:00:00 | London       | 2016:12:11 | 23:00:00     |    24 | DA       |    600 |
| 23 | DA345      | Delta Airlines  | Boston   | 2016:12:12 | 16:00:00 | Atlanta      | 2016:12:12 | 19:00:00     |    50 | DA       |    500 |
| 24 | DA855      | Delta Airlines  | Oakland  | 2016:12:14 | 05:00:00 | Raleigh      | 2016:12:14 | 18:00:00     |    80 | DA       |    500 |
| 25 | BA658      | British Airways | Boston   | 2016:12:12 | 18:00:00 | Phoenix      | 2016:12:12 | 22:00:00     |    80 | BA       |    700 |
+----+------------+-----------------+----------+------------+----------+--------------+------------+--------------+-------+----------+--------+
12 rows in set (0.06 sec)
```

3. customers



```
mysql> select *from customers;
+----+----------+------------+-----------------+----------+------------+----------+------------+------------+--------------+----------+-------+
| ID | username | flight_num | flight_name     | dep_city | dep_date   | dep_time | arivl_city | arivl_date | arrival_time | Bookings | Total |
+----+----------+------------+-----------------+----------+------------+----------+------------+------------+--------------+----------+-------+
|  9 | nikhil   | CP420      | Cathay Pacific  | Dallas   | 2016:12:13 | 07:00:00 | Austin     | 2016:12:13 | 10:00:00     |        1 |   600 |
| 20 | sang     | DA654      | Delta Airlines  | Delhi    | 2016:12:11 | 18:00:00 | London     | 2016:12:11 | 23:00:00     |        1 |   600 |
| 21 | lavanya  | DA654      | Delta Airlines  | Delhi    | 2016:12:11 | 18:00:00 | London     | 2016:12:11 | 23:00:00     |        1 |   600 |
| 22 | kd       | DA654      | Delta Airlines  | Delhi    | 2016:12:11 | 18:00:00 | London     | 2016:12:11 | 23:00:00     |        4 |  2400 |
+----+----------+------------+-----------------+----------+------------+----------+------------+------------+--------------+----------+-------+
4 rows in set (0.00 sec)
```

CONCLUSION

Airline Reservation System has led to ease of airline ticketing, flight scheduling. It has also provided a means for the customers to access and book flight tickets with ease and in time. It has increased the speed with which information about customers and flights are retrieved and handled.

In this project we have successfully designed Airline Reservation System based on most recent version of the Internet Protocol; IPv6 using BSD sockets. We have designed a system which allows customer to book tickets with ease. We have also added few extra special features to the system such as sending confirmation email to the customer after a successful registration and/or ticket booking which allows the customer to know and keep record of his bookings. Another such extra feature is the encryption. It masks the password entered by the users.

REFERENCES

- Internetworking with TCP/IP Vol. 3, Client-Server programming and applications, Comer and Stevens, Linux/POSIX Sockets version, ISBN: 0-13-032071-4, 2001.

- UNIX Network Programming Vol. 1, 3/e: The Sockets Networking API, Stevens, Fenner & Rudoff, ISBN: 0-13-141155-1, 2004.

- UNIX Network Programming Vol. 1, 2/e: Networking APIs - Sockets and XTI, W. Stevens, ISBN: 0-13-490012-X, 1998.

- UNIX Network Programming, Vol. 2, 2/e: Interprocess Communications, W. Stevens, ISBN 0-13-081081-9, 1999.

- UNIX Network Programming, Stevens, ISBN: 0-13-949876-1, 1990.

- https://en.wikipedia.org/wiki/IPv6

- https://en.wikipedia.org/wiki/IPv4

- http://www.informit.com/articles/article.aspx?p=169505&seqNum=2

- https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol

- http://www.lowtek.com/sockets/select.html

- http://www.zdnet.com/article/five-ways-for-ipv6-and-ipv4-to-peacefully-co-exist/

- http://www.networkcomputing.com/networking/six-benefits-ipv6/1148014746

Appendix

Client side code:

Server side code:

SMTP functions for sending mail: