Distributed Airline Reservation System

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. The Airline Representative who can add/delete specific flights, view its flights. And the customers who can make reservations. All the data needed for the application is stored in the form of tables in the MySQL server.
The Airline Reservation System project mainly consists of two. The customers who access the information provided by the website and the administrator who modifies and updates the information available in the website.
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.

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. We use mainly 3 tables to store the information. One table is used to store all the login information. This information is used during the authentication. The information stored in this table is compared against the login details from the users. Second table is used to store all the flight information. It contains the details of all the flights in the database. It has details such as the flight number, flight name, departure date, arrival date etc. Another table is used to store all the information of the customers using the Airline Reservation system. It provides details of their bookings and also their information such as email id.

SMTP:

Features/Implementations:

1. BSD Sockets – Server/Client

2. Database – MySQL

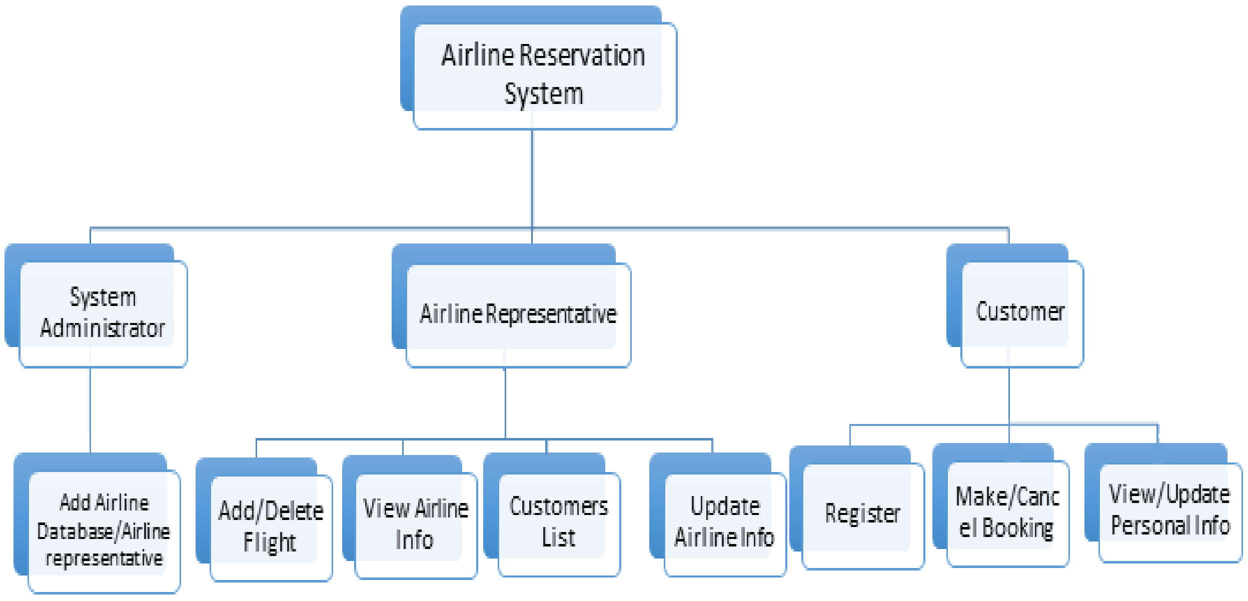
3. Multi-user authentication Login required

Methodology:

There are three main players in this project:

1. System Administrator

2. Airline Representative
3. Customer



The project is designed on the client/server model. We have used BSD sockets for client and server, as it is well integrated with the OS.

The Distributed airline reservation system consists of a client and server program. When the client requests for some information the server fetches the data from the SQL database. There are three main players on the client side. It can be either the System administrator or Airline representative or customer. When any of these users login, the server has to verify the user name and password against the data stored in the database. If the data matches, i.e. after authentication, the user gains access to the database. The server should know who is trying to access the database. Based on the roles, the functionality differs. The server allows the client to either write or read data from the database based on the client's role.

1. System Administrator:

System admin holds the power to add or delete airline representatives. The Administrator also has the sole right to add, delete or modify the flight information. System admin is considered as server which takes login credentials from airline representatives and authenticates them to view the flight details.

When system administrator adds an airline representative, he can access the database. He has the access to the details of his flights and his customers. Daily the Airline Reservation System will have many customers registering with the website and many of them unsubscribing. The airline representative has the rights to modify the information in the database accordingly. He can either add or delete his flights. He can view the airline information, manipulate it and also view his customers details. Whenever any flight information has to be modified or if any new flights need to be added to the database, these operations are performed by the respective airline representative. Sometimes, if any flights get cancelled for some reason, then such flights would be removed from the list of flights available to the customer.

2. Airline Representative:

Airline representative is added by the system administrator. To access the database, Airline representative has to login with username and password. After successful login, he gains access to the database. Airline representative can add his flights or delete his flights. He can also reschedule flights. Also he has the access to all his customers booking details.

3.Customer:

Customers or the end users are the ones who can book a ticket or delete a ticket. They need to first register if they don't have an account to sign in. If the customer has already registered, then he can directly sign in with the username and password used during the registration. The customer can search for flight, query of seats on particular flights, availability dates, arrival and departure time, making query between two locations, providing details on particular flights that from where to where it will travel and what will be it cost and what will be the total; travel duration. Customer can book flight which meets his entered requirements. After booking the ticket he can view his booking details.

Functionality:

The functionality for the project is divided among the users of the Airline Reservation System:

System Administrator functionality:

- Add Airline Representative
- Delete Airline Representative
- View all Airline Representative
- View all Customer Details

Airline Representative functionality:

- Sign in
- View flight details
- Add flight
- Reschedule flight
- Delete flight
- View customer details
- Logout

Customer/End User functionality:

- Sign up/ Register: The Airline Reservation System comes with the customer registration details page, where the customer can enter his details and register. He can also create a username and password. Moreover, he will also be able to modify the registration information in case of a change in his e-mail address or any other information

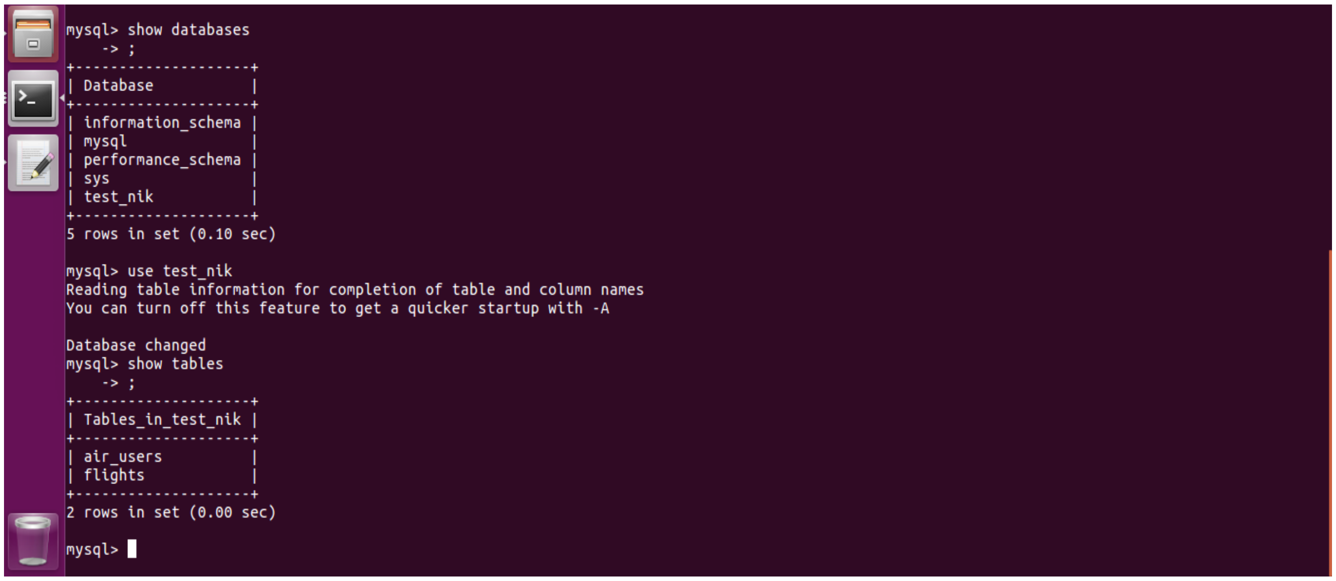
- Login: Customer signs in using the username and password created when registering. After logging in, he can look up information regarding flights. He can now book a ticket or view flights or delete a booking
- View Bookings: In this option, the customer can look up the details of all his bookings.
- Book a flight ticket: The customer can search for the flights available by providing the required details like source, destination, date of journey etc. And reserve his place on the flight by purchasing a ticket.
- Delete bookings: Sometimes, after making a reservation, a customer might want to cancel the reservation he has made. The system allows this change by deleting the selected booking and then updated the customers booking details after removing the booking.
- Logout

SOFTWARE INTERFACE

The application is developed on Linux Operating System. We use MySQL server 2000 Since the application needs a database to store all the customer details, airline, motel and package information, would be used.

Visual Studio.NET 2003 would be used for creating the application. All the coding will be done in C#.

Result:

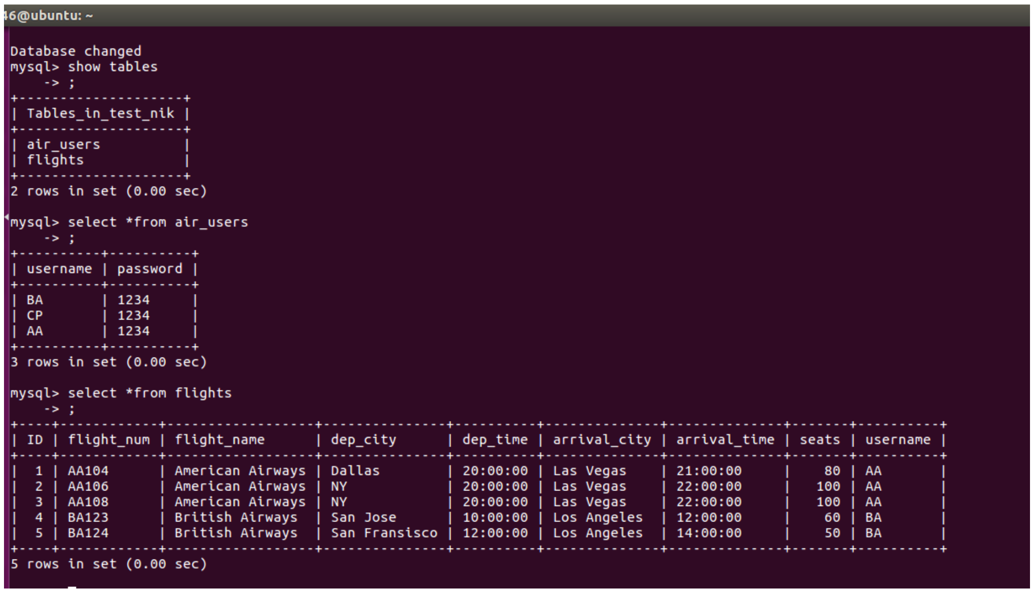


```
mysql> show databases
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test_nik           |
+--------------------+
5 rows in set (0.10 sec)

mysql> use test_nik
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables
    -> ;
+--------------------+
| Tables_in_test_nik |
+--------------------+
| air_users          |
| flights            |
+--------------------+
2 rows in set (0.00 sec)

mysql> 
```



```
46@ubuntu: ~

Database changed
mysql> show tables
    -> ;
+--------------------+
| Tables_in_test_nik |
+--------------------+
| air_users          |
| flights            |
+--------------------+
2 rows in set (0.00 sec)

mysql> select *from air_users
    -> ;
+----------+----------+
| username | password |
+----------+----------+
| BA       | 1234     |
| CP       | 1234     |
| AA       | 1234     |
+----------+----------+
3 rows in set (0.00 sec)

mysql> select *from flights
    -> ;
+----+------------+------------------+---------------+----------+--------------+--------------+-------+----------+
| ID | flight_num | flight_name      | dep_city      | dep_time | arrival_city | arrival_time | seats | username |
+----+------------+------------------+---------------+----------+--------------+--------------+-------+----------+
|  1 | AA104      | American Airways | Dallas        | 20:00:00 | Las Vegas    | 21:00:00     |    80 | AA       |
|  2 | AA106      | American Airways | NY            | 20:00:00 | Las Vegas    | 22:00:00     |   100 | AA       |
|  3 | AA108      | American Airways | NY            | 20:00:00 | Las Vegas    | 22:00:00     |   100 | AA       |
|  4 | BA123      | British Airways  | San Jose      | 10:00:00 | Los Angeles  | 12:00:00     |    60 | BA       |
|  5 | BA124      | British Airways  | San Fransisco | 12:00:00 | Los Angeles  | 14:00:00     |    50 | BA       |
+----+------------+------------------+---------------+----------+--------------+--------------+-------+----------+
5 rows in set (0.00 sec)
```