INTRODUCTION

This project is a Java-based airline management system designed to streamline various tasks related to customer management, ticket booking, flight scheduling, and user administration. The system is built using Java Swing for the graphical user interface (GUI) components and MySQL for database management.

Key Features:

Customer Management: The system allows airline staff to add new customers and search for existing ones, facilitating efficient management of customer information such as names, contact details, and passport information.

Ticket Booking: Users can book tickets for flights through a simple and intuitive interface. The system handles ticket reservations, seat allocations, and generates booking reports for easy tracking.

Flight Management: Staff members can add new flights to the system, specifying details such as flight numbers, departure/arrival times, and destination airports. This feature ensures accurate scheduling and management of flight operations.

User Administration: The system includes user management functionality, allowing administrators to create new user accounts with specific roles and permissions. This ensures proper access control and security within the application.

Technical Details:

Programming Language: Java

GUI Framework: Java Swing

Database Management System: MySQL

IDE: NetBeans

Deployment: The application is deployed locally

UNDERSTANDING AIRLINE MANAGEMENT SYSTEM

Introduction to Airline Management System (AMS):

AMS is a comprehensive software solution designed to streamline various operations within an airline.It encompasses functionalities such as customer management, ticket booking, flight scheduling, and user administration.

Key Components of AMS:

Customer Management: Facilitates the management of customer information, including personal details, contact information, and passport details.Ticket Booking: Allows users to book tickets for flights, manage reservations, and allocate seats efficiently.

Flight Management: Enables the scheduling and management of flight operations, including adding new flights, updating schedules, and tracking flight status.

User Administration: Provides tools for creating and managing user accounts with specific roles and permissions to ensure proper access control.

Benefits of AMS:

Efficiency: Automates manual tasks, reducing errors and saving time for airline staff.

Accuracy: Ensures accurate and up-to-date information regarding customers, flights, and bookings.

Improved Customer Experience: Enhances the booking process, provides timely updates, and improves overall customer satisfaction.

Enhanced Security: Implements access controls and user permissions to protect sensitive data and ensure data integrity.

Technical Aspects:

Programming Language: Developed using Java programming language for its platform independence and robustness.

GUI Framework: Utilizes Java Swing for creating the graphical user interface, offering a rich set of components for building interactive applications.

Database Management: Integrates with MySQL for efficient storage and retrieval of data related to customers, flights, and bookings.

Deployment: Can be deployed locally, depending on the airline's requirements and infrastructure.

Conclusion: The Airline Management System is a crucial tool for modern airlines to streamline operations, enhance efficiency, and provide better services to customers.By leveraging technology, airlines can optimize their processes, improve resource utilization, and ultimately achieve greater success in the competitive aviation industry.

FUNCTIONALITIES OF THIS PROJECT

Add Customer: This module allows users to add new customer information to the system. It involves capturing details such as customer name, passport information, contact details, etc.

Book Ticket: This module facilitates the booking of flight tickets. Users can search for available flights, select seats, specify travel dates, and complete the booking process.

Ticket Report: This module generates a report listing all booked tickets. It provides an overview of ticket information such as ticket number, flight details, customer ID, class, price, seats, and date.

Add Flight: This module enables administrators to add new flights to the system. It involves entering flight details such as flight name, source, destination, departure time, arrival time, and flight charge.

User Creation: This module allows administrators to create new user accounts for the system. It involves collecting user information such as first name, last name, username, and password.

FUNCTIONALITIES DETAILED EXPLANATION IN THE NEXT FOLLOWED PAGES.

LOGIN PAGE

Purpose of Login Page: The login page serves as the gateway for users to access the Airline Management System (AMS).It ensures that only authorized users with valid credentials can log in and perform actions within the system.

User Authentication: The login page verifies the identity of users by requesting their username and password. It validates the provided credentials against the database to authenticate users.

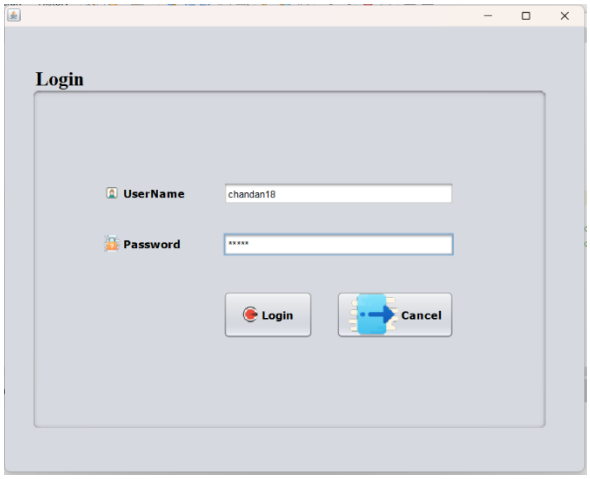
Input Validation: Users are prompted to enter their username and password. Input validation ensures that both fields are filled before proceeding with the login process.If either field is left blank, users are prompted with a message indicating the requirement to fill in both fields.

Connection to Database: Upon entering valid credentials, the login page establishes a connection to the MySQL database. It executes a SQL query to retrieve user information from the database for authentication.

Handling Authentication Results: If the provided credentials match those stored in the database, users are granted access to the main application interface .In case of invalid credentials, users are notified with a message indicating that the username and password do not match.

User Experience: The design of the login page aims to provide a user-friendly experience, with intuitive input fields and clear instructions. Feedback messages inform users about the outcome of their login attempt, guiding them on how to proceed.

Security Measures: The login page implements security measures such as password hashing and secure database connections to protect user data.It plays a crucial role in safeguarding the AMS against unauthorized access and potential security threats.



MAIN PAGE

Navigation Menu: The main page serves as the central hub for navigating different functionalities of the Airline Management System (AMS).It features a user-friendly navigation menu that categorizes various tasks and operations.

Customer Management: Users can access customer management functionalities such as adding new customers and searching for existing ones.The "Customer" menu provides options for adding customers and searching for them based on specific criteria.

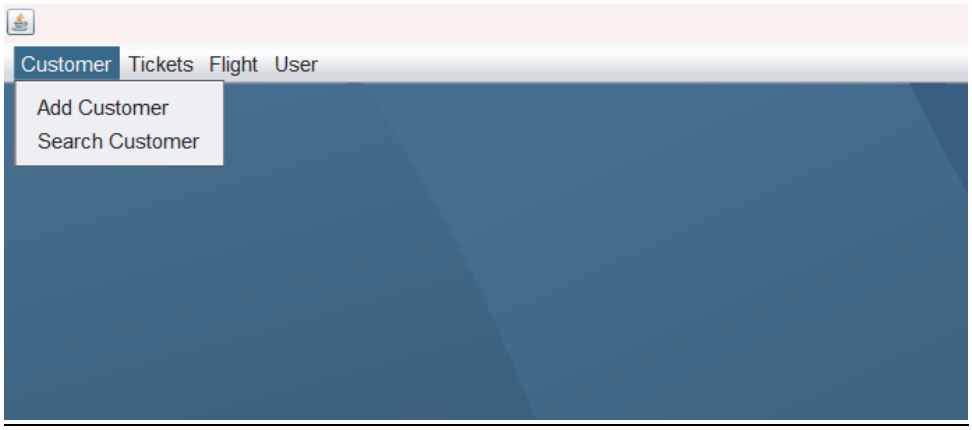
Ticket Operations: Ticket-related operations are conveniently accessible through the "Tickets" menu.Users can book tickets for flights and generate ticket reports for administrative purposes.

Flight Management: The "Flight" menu offers functionalities related to managing flights within the system .Users can add new flights to the system, providing essential details such as flight number, destination, and departure time.

User Administration: User administration tasks, such as creating new user accounts, are available under the "User" menu. Administrators can create user accounts for new employees or system users, defining their access levels and permissions.

Desktop Workspace: The main page utilizes a desktop workspace (JDesktopPane) to host and manage internal frames for different functionalities.Each operation or task opens as an internal frame within the main interface, enhancing organization and usability.

Graphical User Interface (GUI): The graphical user interface (GUI) of the main page is designed for ease of use and intuitive navigation.Visual elements such as icons and labels aid users in quickly identifying and accessing desired functionalities.



ADD CUSTOMER

Form Input Fields: The Add Customer page provides input fields for capturing essential customer information, including first name, last name, NIC number, passport ID, and address. Users can conveniently input customer details through these fields to create new customer records in the system.

Dynamic Customer ID Generation: The system dynamically generates a unique customer ID for each new entry.This ID is automatically incremented based on the existing maximum ID in the database, ensuring uniqueness and consistency in customer identification.

Additional Information: Apart from basic details, users can input additional information such as date of birth, gender, and contact number. Date of birth is captured using a date chooser component for accurate date selection. Gender selection is facilitated through radio buttons for simplicity.

Photo Upload Feature: Users can upload customer photos directly from their devices. The system allows browsing for an image file and displays a preview of the selected photo.

Database Interaction: Upon clicking the "Add" button, the entered customer information is stored in the database. Database interactions are handled using JDBC (Java Database Connectivity) to ensure secure and efficient data management.

User Feedback: The system provides feedback messages, such as "Registration Created," to notify users of successful operations.This feedback enhances user experience and ensures clarity regarding the outcome of their actions.

SEARCH CUSTOMER

User Interface (UI): The UI provides fields to search for a customer by their ID (txtcustid).Upon searching, the customer's details are displayed in various input fields for viewing and potential modification.

Automatic Customer ID Generation: Similar to the "Add Customer" page, the system automatically generates a customer ID (txtcustid) for new records using the autoID() method.

Data Retrieval: Upon entering a customer ID and clicking the "Search" button (jButton4), the system retrieves the corresponding customer record from the database.The retrieved data includes the customer's first name, last name, NIC number, passport ID, address, date of birth, gender, contact information, and photo.

Photo Display: If a customer photo is available in the database, it is displayed in the designated area (txtphoto) using the Blob object to handle image data.

Data Population: The retrieved customer data is populated into their respective input fields (txtfirstname, txtlastname, txtnic, txtpassport, txtaddress, txtdob, r1, r2, txtcontact) for viewing and potential editing.

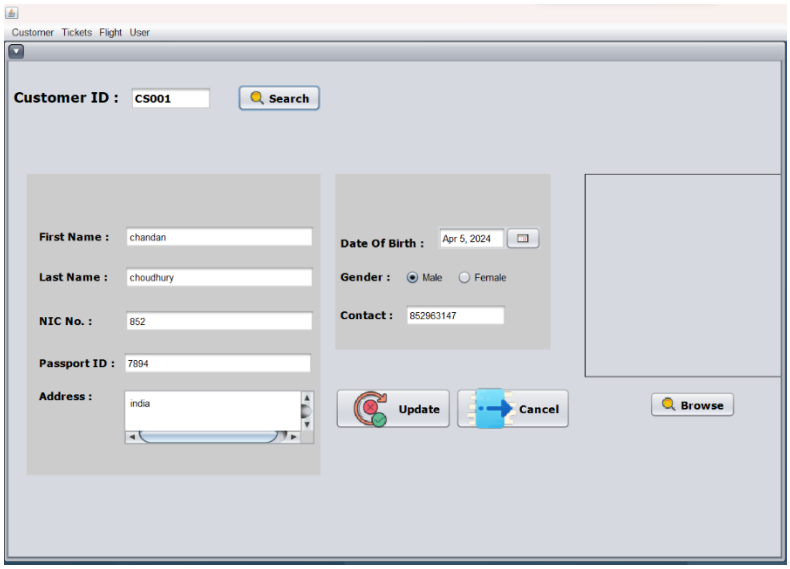
Gender Selection: The customer's gender is displayed and selected using radio buttons (r1, r2), with the appropriate radio button being automatically selected based on the retrieved data.

Update Functionality: Users have the option to update the customer's information by making changes in the displayed fields and clicking the "Update" button (jButton2).

Cancel Functionality: Users can cancel the operation and close the "Search Customer" window by clicking the "Cancel" button (jButton3).

Exception Handling: The system includes exception handling to address potential errors during data retrieval or processing, such as SQL exceptions, null pointer exceptions, or parsing errors.

Overall Functionality: The "Search Customer" page enhances the efficiency of customer management by providing a convenient way to retrieve and view customer details from the database, facilitating customer service and record maintenance.



TICKET BOOKING

User Interface (UI): The UI consists of various input fields and components to facilitate the booking process. Users can select the source and destination airports (txtsource, txtdepart), as well as the date of travel (txtdate).Flight options meeting the selected criteria are displayed in a table (jTable1), allowing users to choose their desired flight.

Flight Search: Upon selecting the source and destination airports and clicking the "Search" button (jButton3), the system retrieves available flights from the database.The retrieved flight details include the flight number, flight name, source, destination, date, departure time, arrival time, and flight charge.

Customer Information Retrieval: Users can enter the customer ID (txtcustid) and click the "Search" button (jButton4) to retrieve the customer's details from the database.The retrieved customer information includes the first name, last name, and passport ID, which are automatically populated in the corresponding fields.

Flight Selection: Users can click on a row in the flight table (jTable1) to select a specific flight for booking.Upon selection, the flight details, including flight number, flight name, departure time, and price, are displayed in designated fields (flightno, flightname, txtdept, txtprice).

Seat Selection: Users can use a spinner (txtseats) to select the number of seats they wish to book.The total price for the selected number of seats is automatically calculated and displayed (txttotal).

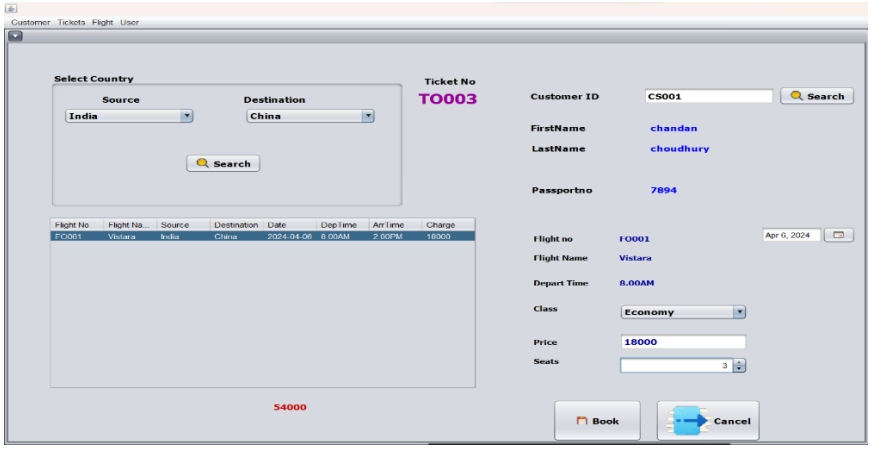
Ticket Booking: Upon entering all necessary information and clicking the "Book Ticket" button (jButton1), the system records the ticket booking in the database.The ticket details, including the ticket ID, flight ID, customer ID, class, price, number of seats, and date of travel, are stored in the database.

Automatic Ticket ID Generation: Similar to the "Add Customer" page, the system automatically generates a ticket ID (txtticketno) for new ticket bookings using the autoID() method.

Cancel Functionality: Users can cancel the booking process and close the "Book Ticket" window by clicking the "Cancel" button (jButton2).

Exception Handling: The system includes exception handling to address potential errors during data retrieval or processing, such as SQL exceptions or class not found exceptions.

Overall Functionality: The "Book Ticket" page streamlines the ticket booking process, allowing users to search for flights, select seats, and book tickets efficiently while ensuring the integrity of customer and flight data in the database.



TICKET REPORT

User Interface (UI): The UI consists of a table (jTable1) that displays a list of ticket details. Users can view information such as ticket number, flight number, customer ID, class, price, number of seats, and date of travel.

Data Loading: Upon initialization, the page automatically loads ticket data from the database using the LoadData() method. The LoadData() method retrieves ticket information from the database table and populates the table with the retrieved data.

Table Display: The table is dynamically populated with ticket information retrieved from the database. Each row in the table represents a single ticket, displaying relevant details such as ticket number, flight number, customer ID, etc.

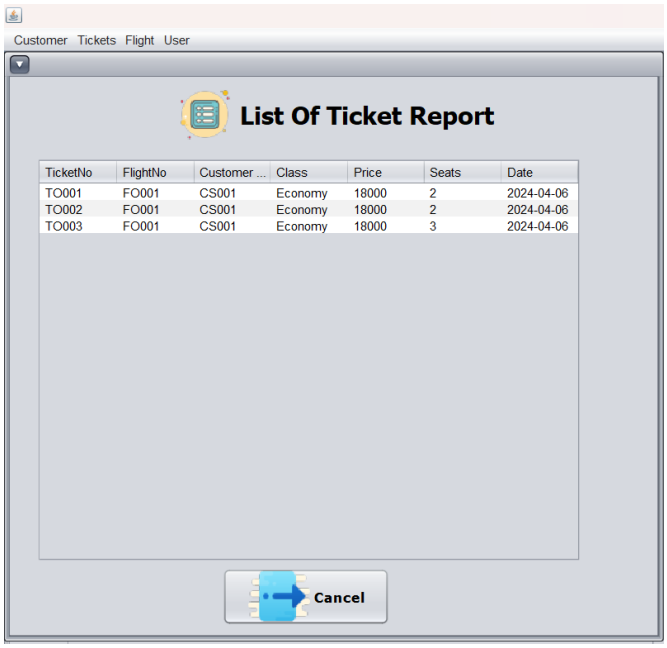
Data Retrieval: Ticket information is retrieved from the database using a SQL query (select * from ticket).The retrieved data is stored in a ResultSet object, and its metadata is obtained to determine the number of columns.

TableModel Management: The DefaultTableModel (Df) is used to manage the data displayed in the table.Rows are added to the table model (Df) dynamically as ticket information is retrieved from the database.

Cancel Functionality: Users can close the "Ticket Report" window by clicking the "Cancel" button (jButton1).

Exception Handling: The system includes exception handling to address potential errors during data retrieval or processing, such as SQL exceptions or class not found exceptions.

Overall Functionality: The "Ticket Report" page provides users with a comprehensive view of all ticket bookings made in the system. Users can easily review ticket details, facilitating administrative tasks and reporting requirements.



ADD FLIGHT

User Interface (UI): The UI consists of input fields and dropdown menus allowing users to enter flight details such as flight ID, flight name, source, destination, date, departure time, arrival time, and flight charge.

Automatic ID Generation: The system automatically generates a unique flight ID (txtflightid) for each new flight added. This functionality is implemented using the autoID() method, which queries the database to determine the maximum flight ID and increments it to generate the next ID.

Data Entry: Users can input flight details into the respective fields, including the flight name, source (departure location), destination (departure location), date of departure, departure time, arrival time, and flight charge.

Date Selection: The date of departure is selected using a date chooser (txtdate), ensuring accurate input format and ease of use for users.

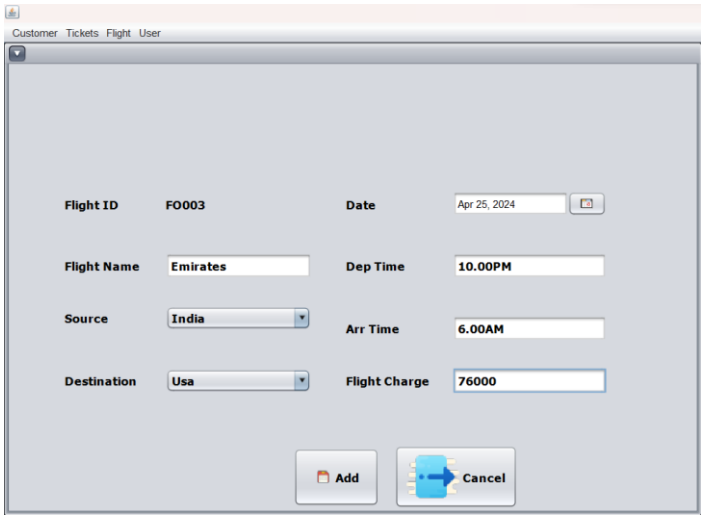
Data Validation: Input data is validated to ensure accuracy and completeness before being inserted into the database.Validation includes checking for empty fields and verifying the format of input data such as dates and times.

Database Interaction: Upon clicking the "Create" button (jButton1), the entered flight details are inserted into the database table flight. Database interaction is handled using JDBC (Java Database Connectivity) with MySQL, ensuring seamless communication between the application and the database.

Exception Handling: The system includes exception handling to address potential errors during database interaction, such as SQL exceptions or class not found exceptions. Error messages are displayed to the user using JOptionPane in case of database errors.

Cancellation: Users can cancel the operation and close the "Add Flight" window by clicking the "Cancel" button (jButton2).

Overall Functionality: The "Add Flight" page provides a user-friendly interface for adding new flights to the system, enabling efficient management of flight data for airline administrators.



USER CREATION

User Interface (UI): The UI consists of input fields allowing administrators to create new user accounts.Input fields include first name, last name, username, and password.

Automatic ID Generation: Similar to other modules, this page also automatically generates a unique user ID (txtuserid) for each new user created. The autoID() method queries the database to determine the maximum user ID and increments it to generate the next ID.

Data Entry: Administrators can input user details into the respective fields, including first name, last name, username, and password.

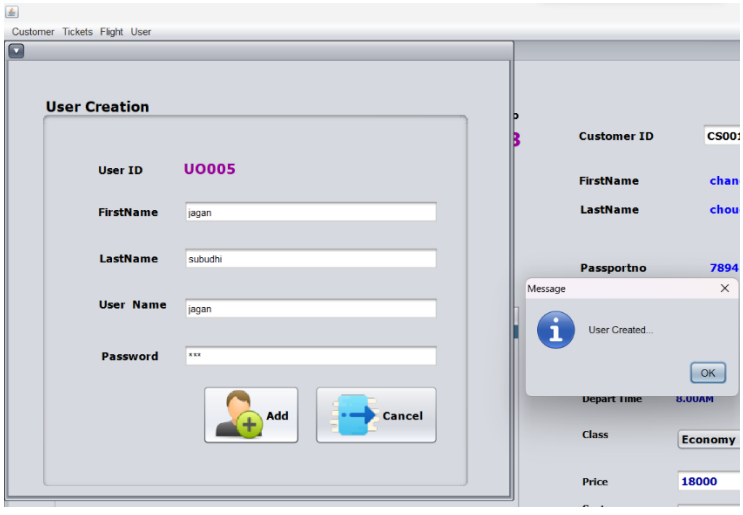
Data Validation: Input data is validated to ensure accuracy and completeness before being inserted into the database. Validation includes checking for empty fields and ensuring secure password entry.

Database Interaction: Upon clicking the "Add" button (jButton1), the entered user details are inserted into the database table user. Database interaction is handled using JDBC (Java Database Connectivity) with MySQL, ensuring seamless communication between the application and the database.

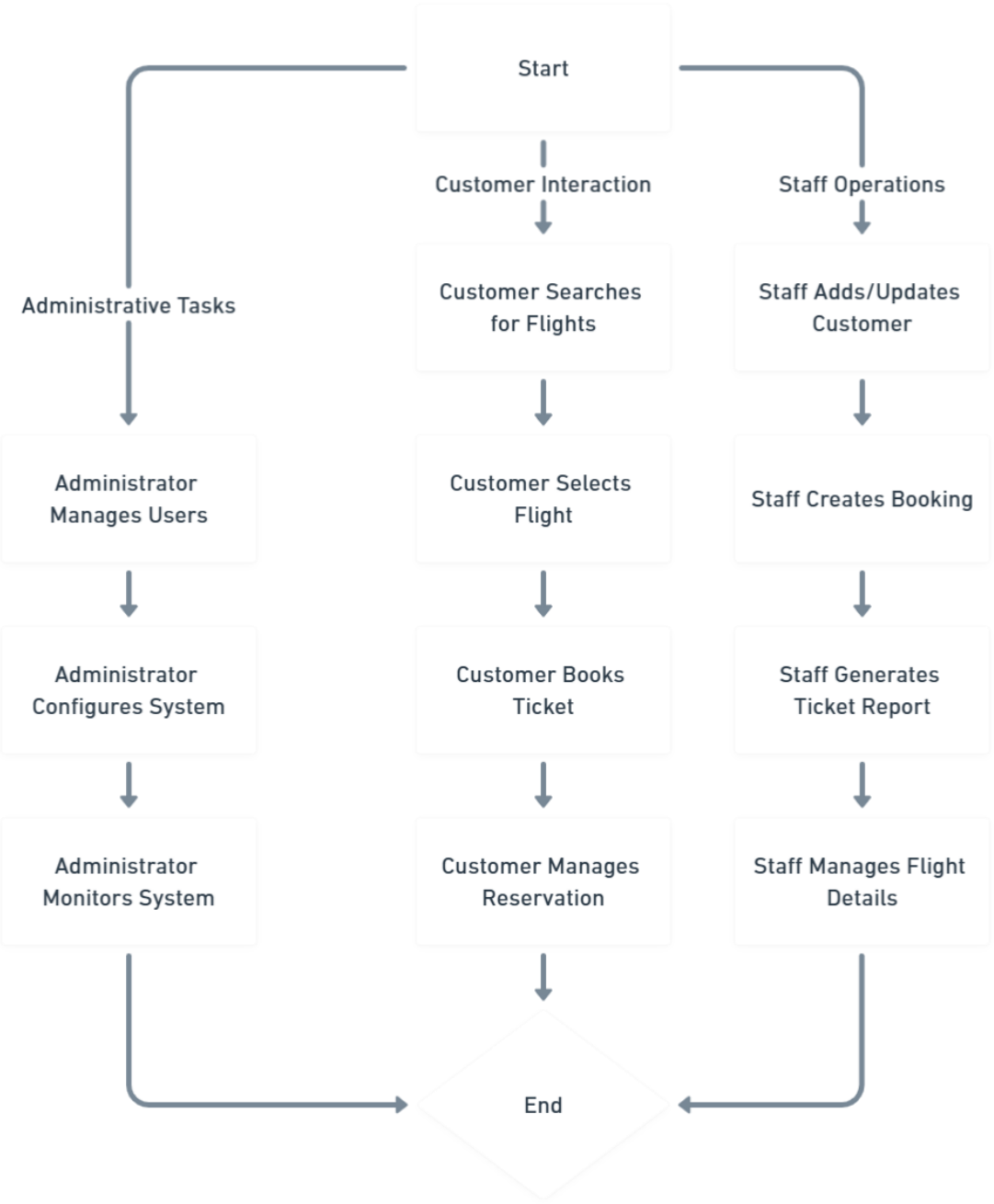
Exception Handling: The system includes exception handling to address potential errors during database interaction, such as SQL exceptions or class not found exceptions. Error messages are displayed to the administrator using JOptionPane in case of database errors.

Cancellation: Administrators can cancel the operation and close the "User Creation" window by clicking the "Cancel" button (jButton2).

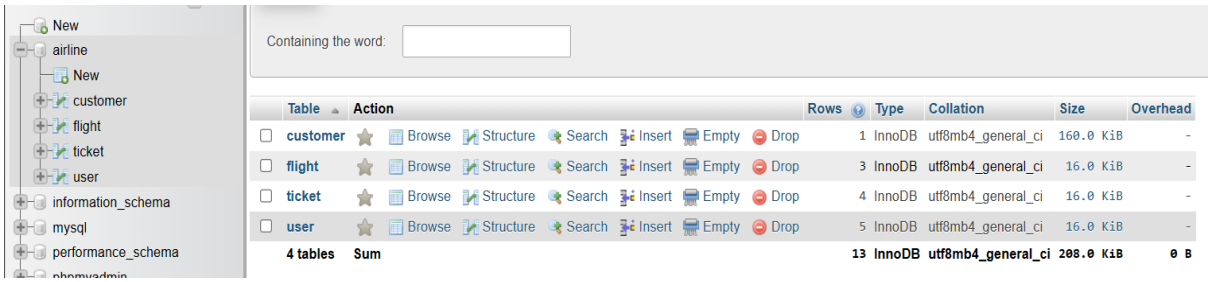
Overall Functionality: The "User Creation" page provides a user-friendly interface for administrators to add new users to the system, enabling efficient management of user accounts for the airline application.



FLOWCHART


Start
Customer Interaction
Staff Operations
Administrative Tasks
Customer Searches for Flights
Staff Adds/Updates Customer
Administrator Manages Users
Customer Selects Flight
Staff Creates Booking
Administrator Configures System
Customer Books Ticket
Staff Generates Ticket Report
Administrator Monitors System
Customer Manages Reservation
Staff Manages Flight Details
End

DATABASE AND TABLES USED



Database airline and the Tables in it.

Customer table structure and table

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
2	firstname	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	lastname	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	nic	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	passport	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
6	address	text	utf8mb4_general_ci		No	None			Change Drop More
7	dob	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
8	gender	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
9	contact	int(11)			No	None			Change Drop More
10	photo	longblob			No	None			Change Drop More

id	firstname	lastname	nic	passport	address	dob	gender	contact	photo
CS001	chandan	choudhury	852	7894	india	2024-04-05	Male	852963147	[BLOB - 245.4 KiB]

Ticket table structure and table

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
2	flightid	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	custid	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	class	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	price	int(11)			No	None			Change Drop More
6	seats	int(11)			No	None			Change Drop More
7	date	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

id	flightid	custid	class	price	seats	date
TO001	FO001	CS001	Economy	18000	2	2024-04-06
TO002	FO001	CS001	Economy	18000	2	2024-04-06
TO003	FO001	CS001	Economy	18000	3	2024-04-06
TO004	FO001	CS001	Economy	18000	3	2024-04-06

Flight table structure and table

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
2	flightname	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	source	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	depart	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	date	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
6	deptime	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
7	arrtime	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
8	flightcharge	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

id	flightname	source	depart	date	deptime	arrtime	flightcharge
FO001	Vistara	India	China	2024-04-06	8.00AM	2.00PM	18000
FO002	Etihad	India	Usa	2024-04-25	8.30AM	10.30PM	70000
FO003	Emirates	India	Usa	2024-04-25	10.00PM	6.00AM	76000

User table structure and table

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
2	firstname	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	lastname	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	username	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

id	firstname	lastname	username	password
UO001	chandan	choudhury	chandan	352
UO002	aditya	nayak	aditya	377
UO003	aditya	sinha	aditya378	378
UO004	chandan	choudhury	chandan18	35218
UO005	jagan	subudhi	jagan	364

CODE

Login page code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import java.sql.ResultSet;
public class Login extends javax.swing.JFrame {
    public Login() {
        initComponents();
    }
    Connection con;
    PreparedStatement pst;
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String username = txtuser.getText();
        String password= txtpass.getText();
        if(username.isEmpty() || password.isEmpty()){
            JOptionPane.showMessageDialog(this,"UserName or Password is Blank");
        }
        else{
            try {
                Class.forName("com.mysql.cj.jdbc.Driver");
                con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
                pst = con.prepareStatement("SELECT * FROM user WHERE username = ? AND password = ?");
                pst.setString(1, username);
                pst.setString(2, password);
                ResultSet rs;
                rs = pst.executeQuery();
                if(rs.next())
                {
```

```
                Main m = new Main();
                this.hide();
                m.setVisible(true);
            }else
            {
                JOptionPane.showMessageDialog(this,"UserName and Password do not Match");
                txtuser.setText("");
                txtpass.setText("");
                txtuser.requestFocus();
            }
        } catch (ClassNotFoundException | SQLException ex) {
            Logger.getLogger(Login.class.getName()).log(Level.SEVERE, null, ex);
        }  }}
    public static void main(String args[]) {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }  }} catch (ClassNotFoundException | InstantiationException | IllegalAccessException |
javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Login().setVisible(true);
            });}}
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPasswordField txtpass;
    private javax.swing.JTextField txtuser;
    // End of variables declaration
}
```

Main page code:

```
public class Main extends javax.swing.JFrame {
    public Main() {
        initComponents();
    }
    private void initComponents() {
        jDesktopPane1 = new javax.swing.JDesktopPane();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenuItem2 = new javax.swing.JMenuItem();
        jMenu2 = new javax.swing.JMenu();
        jMenuItem3 = new javax.swing.JMenuItem();
        jMenuItem6 = new javax.swing.JMenuItem();
        jMenu3 = new javax.swing.JMenu();
        jMenuItem4 = new javax.swing.JMenuItem();
        jMenu4 = new javax.swing.JMenu();
        jMenuItem5 = new javax.swing.JMenuItem();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        jMenu1.setText("Customer");
        jMenuItem1.setText("Add Customer");
        jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }});
        jMenu1.add(jMenuItem1);
        jMenuItem2.setText("Search Customer");
        jMenuItem2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem2ActionPerformed(evt);
            }});
        jMenu1.add(jMenuItem2);
        jMenuBar1.add(jMenu1);
        jMenu2.setText("Tickets");
jMenuItem3.setText("Book Ticket");
```

```
jMenuItem3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem3ActionPerformed(evt);
    }});
jMenu2.add(jMenuItem3);
jMenuItem6.setText("Ticket Report");
jMenuItem6.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem6ActionPerformed(evt);
    }});
jMenu2.add(jMenuItem6);
jMenuBar1.add(jMenu2);
jMenu3.setText("Flight");
jMenuItem4.setText("Add Flight");
jMenuItem4.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem4ActionPerformed(evt);
    }});
jMenu3.add(jMenuItem4);
jMenuBar1.add(jMenu3);
jMenu4.setText("User");
jMenuItem5.setText("UserCreation");
jMenuItem5.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem5ActionPerformed(evt); }});
jMenu4.add(jMenuItem5);
jMenuBar1.add(jMenu4);
setJMenuBar(jMenuBar1);
javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jDesktopPane1));
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jDesktopPane1));
```

```
        pack();}
    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
        addCustomer cus = new addCustomer();
        jDesktopPane1.add(cus);
        cus.setVisible(true);}
    private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
        searchCustomer scus = new searchCustomer();
        jDesktopPane1.add(scus);
        scus.setVisible(true);}
    private void jMenuItem4ActionPerformed(java.awt.event.ActionEvent evt) {
        addflight f = new addflight();
        jDesktopPane1.add(f);
        f.setVisible(true);}
    private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
        ticket t = new ticket();
        jDesktopPane1.add(t);
        t.setVisible(true);
    }
    private void jMenuItem6ActionPerformed(java.awt.event.ActionEvent evt) {
        ticketreport ti = new ticketreport();
        jDesktopPane1.add(ti);
        ti.setVisible(true);            }
    private void jMenuItem5ActionPerformed(java.awt.event.ActionEvent evt) {
        userCreation u = new userCreation();
        jDesktopPane1.add(u);
        u.setVisible(true);
    }
    public static void main(String args[]) {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }}
```

```
        } catch (ClassNotFoundException | InstantiationException | IllegalAccessException |
javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);}
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Main().setVisible(true);
            }});}
    private javax.swing.JDesktopPane jDesktopPane1;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenu jMenu2;
    private javax.swing.JMenu jMenu3;
    private javax.swing.JMenu jMenu4;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax
```

Add Customer page code:

```
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileNameExtensionFilter;
import java.sql.PreparedStatement;
public class addCustomer extends javax.swing.JInternalFrame {
    public addCustomer() {
        initComponents();
        autoID();}
    Connection con;
     PreparedStatement pst;
    String path = null;
    byte[] userimage=null;
    public void autoID() {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
        Statement s = con.createStatement();
```

```
        ResultSet rs = s.executeQuery("select MAX(id) from customer");
        rs.next();
        String maxId = rs.getString("MAX(id)");
        if (maxId == null) {
           txtid.setText("CS001");
        } else {
           long id = Long.parseLong(maxId.substring(2)) + 1;
           txtid.setText("CS" + String.format("%03d", id));
        }} catch (ClassNotFoundException | SQLException ex) {
        Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
    }}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            JFileChooser picchooser= new JFileChooser();
        picchooser.showOpenDialog(null);
        File pic = picchooser.getSelectedFile();
        FileNameExtensionFilter filter = new FileNameExtensionFilter("*.images","png","jpg");
        picchooser.addChoosableFileFilter(filter);
        path=pic.getAbsolutePath();
        BufferedImage img;
        img = ImageIO.read(picchooser.getSelectedFile());
        ImageIcon imageIcon = new ImageIcon(new
        ImageIcon(img).getImage().getScaledInstance(250,250,Image.SCALE_DEFAULT));
        txtphoto.setIcon(imageIcon);
        File image =new File(path);
        FileInputStream fis = new FileInputStream(image);
        ByteArrayOutputStream baos=new ByteArrayOutputStream();
        byte[] buff = new byte[1024];
        int readNum;
        while ((readNum = fis.read(buff)) != -1) {
        baos.write(buff, 0, readNum);}
        userimage=baos.toByteArray();
        } catch (IOException ex) {
           Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
        } }
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        // TODO add your handling code here:
        String id= txtid.getText();
        String firstname= txtfirstname.getText();
        String lastname= txtlastname.getText();
        String nic= txtnic.getText();
        String passport= txtpassport.getText();
        String address= txtaddress.getText();
        DateFormat da = new SimpleDateFormat("yyyy-MM-dd");
        String date=da.format(txtdob.getDate());
        String Gender;
        if(r1.isSelected()){
           Gender="Male";}
        else{
           Gender="Female";}
        String contact= txtcontact.getText();
        try {
           Class.forName("com.mysql.cj.jdbc.Driver");
           Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
           pst = con.prepareStatement("insert into
customer(id,firstname,lastname,nic,passport,address,dob,gender,contact,photo)values(?,?,?,?,?,?,?,?,?,?)");
           pst.setString(1,id);
           pst.setString(2,firstname);
           pst.setString(3,lastname);
           pst.setString(4,nic);
           pst.setString(5,passport);
           pst.setString(6,address);
           pst.setString(7,date);
           pst.setString(8,Gender);
           pst.setString(9,contact);
           pst.setBytes(10,userimage);
           pst.executeUpdate();
           JOptionPane.showMessageDialog(null,"Registration Created...");
        } catch (ClassNotFoundException ex) {
           Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
           Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
```

```
    }    }
    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        this.hide(); }
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JRadioButton r1;
    private javax.swing.JRadioButton r2;
    private javax.swing.JTextArea txtaddress;
    private javax.swing.JTextField txtcontact;
    private com.toedter.calendar.JDateChooser txtdob;
    private javax.swing.JTextField txtfirstname;
    private javax.swing.JLabel txtid;
    private javax.swing.JTextField txtlastname;
    private javax.swing.JTextField txtnic;
    private javax.swing.JTextField txtpassport;
    private javax.swing.JLabel txtphoto;
}
```

Search Customer page code:

```
import com.mysql.cj.jdbc.Blob;
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileNameExtensionFilter;
import java.sql.PreparedStatement;
import javax.swing.JLabel;
public class searchCustomer extends javax.swing.JInternalFrame {
    Connection con;
    PreparedStatement pst;
    String path = null;
    byte[] userimage = null;
    public searchCustomer() {
        initComponents();
        autoID();}
```

```
    public void autoID() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            Statement s = con.createStatement();
            ResultSet rs = s.executeQuery("select MAX(id) from customer");
            rs.next();
            String maxId = rs.getString("MAX(id)");
            if (maxId == null) {
                txtcustid.setText("CS001");
            } else {
                long id = Long.parseLong(maxId.substring(2)) + 1;
                txtcustid.setText("CS" + String.format("%03d", id));}
        } catch (ClassNotFoundException | SQLException ex) {
            Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);}}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
             JFileChooser picchooser= new JFileChooser();
        picchooser.showOpenDialog(null);
        File pic = picchooser.getSelectedFile();
        FileNameExtensionFilter filter = new FileNameExtensionFilter("*.images","png","jpg");
        picchooser.addChoosableFileFilter(filter);
        path=pic.getAbsolutePath();
        BufferedImage img;
        img = ImageIO.read(picchooser.getSelectedFile());
        ImageIcon imageIcon = new ImageIcon(new
        ImageIcon(img).getImage().getScaledInstance(250,250,Image.SCALE_DEFAULT));
        txtphoto.setIcon(imageIcon);
        File image =new File(path);
        FileInputStream fis = new FileInputStream(image);
        ByteArrayOutputStream baos=new ByteArrayOutputStream();
        byte[] buff = new byte[1024];
        int readNum;
        while ((readNum = fis.read(buff)) != -1) {
        baos.write(buff, 0, readNum);}
        userimage=baos.toByteArray();
```

```
        } catch (IOException ex) {
            Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);}  }
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        String id= txtcustid.getText();
        String firstname= txtfirstname.getText();
        String lastname= txtlastname.getText();
        String nic= txtnic.getText();
        String passport= txtpassport.getText();
        String address= txtaddress.getText();
        DateFormat da = new SimpleDateFormat("yyyy-MM-dd");
        String date=da.format(txtdob.getDate());
        String Gender;
            if(r1.isSelected()){
            Gender="Male";}
        else{
            Gender="Female";}
        String contact= txtcontact.getText();try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst = con.prepareStatement("update customer set
firstname=?,lastname=?,nic=?,passport=?,address=?,dob=?,gender=?,contact=?,photo=? where id = ?");
            pst.setString(1,firstname);
            pst.setString(2,lastname);
            pst.setString(3,nic);
            pst.setString(4,passport);
            pst.setString(5,address);
            pst.setString(6,date);
            pst.setString(7,Gender);
            pst.setString(8,contact);
            pst.setBytes(9,userimage);
            pst.setString(10, id);
            pst.executeUpdate();
            JOptionPane.showMessageDialog(null,"Registration Updated...");
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
```

```
            Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
        }}
    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        this.hide();}
    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        String id = txtcustid.getText();
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst = con.prepareStatement("select*from customer where id=?");
            pst.setString(1, id);
            ResultSet rs=pst.executeQuery();
            if(rs.next()==false){
                JOptionPane.showMessageDialog(this, "Record not Found");}
            else{
                String fname = rs.getString("firstname");
                String lname = rs.getString("lastname");
                String nic = rs.getString("nic");
                String passport = rs.getString("passport");
                String address = rs.getString("address");
                String dob = rs.getString("dob");
                Date date1 =new SimpleDateFormat("yyyy-MM-dd").parse(dob);
                String gender = rs.getString("gender");
                try {
    Blob blob = (Blob) rs.getBlob("photo");
    if (blob != null) {
        byte[] imageData = blob.getBytes(1, (int) blob.length());
        ImageIcon image = new ImageIcon(imageData);
        Image im = image.getImage();
        Image myImg = im.getScaledInstance(txtphoto.getWidth(), txtphoto.getHeight(),
Image.SCALE_SMOOTH);
        ImageIcon newImage = new ImageIcon(myImg);
        txtphoto.setIcon(newImage);
    } else {// Handle the case where the blob is null (no image data)
    }
} catch (SQLException | NullPointerException ex) {
```

```
ex.printStackTrace();}
if(gender.equals("Female")){
                r1.setSelected(false);
                r2.setSelected(true);}
            else{
                r1.setSelected(true);
                r2.setSelected(false);}
            String contact = rs.getString("contact");
            txtfirstname.setText(fname.trim());
            txtlastname.setText(lname.trim());
            txtnic.setText(nic.trim());
            txtpassport.setText(passport.trim());
            txtaddress.setText(address.trim());
            txtcontact.setText(contact.trim());
            txtdob.setDate(date1);
            Icon newImage = null;
            txtphoto.setIcon(newImage);
}} catch (ClassNotFoundException ex) {
            Logger.getLogger(searchCustomer.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
            Logger.getLogger(searchCustomer.class.getName()).log(Level.SEVERE, null, ex);
        } catch (ParseException ex) {
            Logger.getLogger(searchCustomer.class.getName()).log(Level.SEVERE, null, ex);
        }}
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel8;
```

```
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JRadioButton r1;
    private javax.swing.JRadioButton r2;
    private javax.swing.JTextArea txtaddress;
    private javax.swing.JTextField txtcontact;
    private javax.swing.JTextField txtcustid;
    private com.toedter.calendar.JDateChooser txtdob;
    private javax.swing.JTextField txtfirstname;
    private javax.swing.JTextField txtlastname;
    private javax.swing.JTextField txtnic;
    private javax.swing.JTextField txtpassport;
    private javax.swing.JLabel txtphoto;
}
```

Book Ticket page code:

```
import com.mysql.cj.jdbc.Blob;
import com.sun.jdi.connect.spi.Connection;
import java.awt.Image;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Vector;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class ticket extends javax.swing.JInternalFrame {
    public ticket() {
        initComponents();
        autoID();}
    java.sql.Connection con;
     PreparedStatement pst;
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        String source = txtsource.getSelectedItem().toString();
        String depart = txtdepart.getSelectedItem().toString();
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst= con.prepareStatement("select * from flight where source = ? and depart = ?");
            pst.setString(1, source);
            pst.setString(2, depart);
```

```
        ResultSet rs = pst.executeQuery();
        ResultSetMetaData rsm = rs.getMetaData();
        int  c;
        c = rsm.getColumnCount();
        DefaultTableModel Df =(DefaultTableModel)jTable1.getModel();
        Df.setRowCount(0);
while(rs.next())
        {
           Vector v2 = new Vector();
           for(int i= 1; i<=c ; i++)
           {
              v2.add(rs.getString("id"));
              v2.add(rs.getString("flightname"));
              v2.add(rs.getString("source"));
              v2.add(rs.getString("depart"));
              v2.add(rs.getString("date"));
              v2.add(rs.getString("deptime"));
              v2.add(rs.getString("arrtime"));
              v2.add(rs.getString("flightcharge"));
           }
           Df.addRow(v2);
        }       } catch (ClassNotFoundException ex) {
        Logger.getLogger(ticket.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(ticket.class.getName()).log(Level.SEVERE, null, ex);
    }}
 public void autoID() {
 try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    java.sql.Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
    Statement s = con.createStatement();
    ResultSet rs = s.executeQuery("select MAX(id) from ticket");
    rs.next();
    String maxId = rs.getString("MAX(id)");
    if (maxId == null) {
       txtticketno.setText("TO001");
```

```
        } else {
            long id = Long.parseLong(maxId.substring(2)) + 1;
            txtticketno.setText("TO" + String.format("%03d", id));
        }} catch (ClassNotFoundException | SQLException ex) {
        Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
    }}
    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        String id = txtcustid.getText();
        try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    java.sql.Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
    PreparedStatement pst = con.prepareStatement("SELECT * FROM customer WHERE id=?");
    pst.setString(1, id);
    ResultSet rs = pst.executeQuery();
    if (!rs.next()) {
        JOptionPane.showMessageDialog(this, "Record not found");
    } else {
        String fname = rs.getString("firstname");
        String lname = rs.getString("lastname");
        String passport = rs.getString("passport");
        txtfirstname.setText(fname.trim());
        txtlastname.setText(lname.trim());
        txtpassport.setText(passport.trim());}}
 catch (ClassNotFoundException | SQLException ex) {
    Logger.getLogger(ticket.class.getName()).log(Level.SEVERE, null, ex);
}}
    private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {
        DefaultTableModel Df =(DefaultTableModel)jTable1.getModel();
        int selectIndex = jTable1.getSelectedRow();
        flightno.setText(Df.getValueAt(selectIndex, 0).toString());
        flightname.setText(Df.getValueAt(selectIndex, 1).toString());
        txtdept.setText(Df.getValueAt(selectIndex, 5).toString());
        txtprice.setText(Df.getValueAt(selectIndex, 7).toString());
    }
    private void txtseatsStateChanged(javax.swing.event.ChangeEvent evt) {
        int price = Integer.parseInt(txtprice.getText());
```

```
        int qty = Integer.parseInt(txtseats.getValue().toString());
        int tot = price * qty ;
        txttotal.setText(String.valueOf(tot));
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String ticketid = txtticketno.getText();
        String flightid = flightno.getText();
        String custid = txtcustid.getText();
        String flightclass = txtclass.getSelectedItem().toString().trim();
        String price =txtprice.getText();
        String seats = txtseats.getValue().toString();
        DateFormat da = new SimpleDateFormat("yyyy-MM-dd");
        String date = da.format(txtdate.getDate());
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
             con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst = con.prepareStatement("insert into
ticket(id,flightid,custid,class,price,seats,date)values(?,?,?,?,?,?,?)");
            pst.setString(1,ticketid);
            pst.setString(2,flightid);
            pst.setString(3,custid);
            pst.setString(4,flightclass);
            pst.setString(5,price);
            pst.setString(6,seats);
            pst.setString(7,date);
            pst.executeUpdate();
            JOptionPane.showMessageDialog(null,"Ticket Bookeed.....");
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(addflight.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
            Logger.getLogger(addflight.class.getName()).log(Level.SEVERE, null, ex);}}
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        this.hide();
    }
    private javax.swing.JLabel flightname;
    private javax.swing.JLabel flightno;
```

```
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel16;
    private javax.swing.JLabel jLabel17;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable jTable1;
    private javax.swing.JComboBox<String> txtclass;
    private javax.swing.JTextField txtcustid;
    private com.toedter.calendar.JDateChooser txtdate;
    private javax.swing.JComboBox<String> txtdepart;
    private javax.swing.JLabel txtdept;
    private javax.swing.JLabel txtfirstname;
    private javax.swing.JLabel txtlastname;
    private javax.swing.JLabel txtpassport;
    private javax.swing.JTextField txtprice;
    private javax.swing.JSpinner txtseats;
    private javax.swing.JComboBox<String> txtsource;
    private javax.swing.JLabel txtticketno;
    private javax.swing.JLabel txttotal;
}
```

Ticket Report page code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.table.DefaultTableModel;
public class ticketreport extends javax.swing.JInternalFrame {
    public ticketreport() {
        initComponents();
        LoadData();}
    Connection con;
     PreparedStatement pst;
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        this.hide();}
public void LoadData(){
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst= con.prepareStatement("select * from ticket ");
            ResultSet rs = pst.executeQuery();
            ResultSetMetaData rsm = rs.getMetaData();
            int  c;
            c = rsm.getColumnCount();
            DefaultTableModel Df =(DefaultTableModel)jTable1.getModel();
            Df.setRowCount(0);
   while(rs.next())
            {
                Vector v2 = new Vector();
                for(int i= 1; i<=c ; i++)
                {
```

```
                v2.add(rs.getString("id"));
                v2.add(rs.getString("flightid"));
                v2.add(rs.getString("custid"));
                v2.add(rs.getString("class"));
                v2.add(rs.getString("price"));
                v2.add(rs.getString("seats"));
                v2.add(rs.getString("date"));
            }
Df.addRow(v2);
}
}
 catch (ClassNotFoundException ex) {
            Logger.getLogger(ticket.class.getName()).log(Level.SEVERE, null, ex);
        }
catch (SQLException ex) {
            Logger.getLogger(ticket.class.getName()).log(Level.SEVERE, null, ex);
        }
}
```

Add Flight page code:

```
 import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
public class addflight extends javax.swing.JInternalFrame  {
public addflight() {
        initComponents();
        autoID();}
    Connection con;
     PreparedStatement pst;
public void autoID() {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
        Statement s = con.createStatement();
        ResultSet rs = s.executeQuery("select MAX(id) from flight");
        rs.next();
        String maxId = rs.getString("MAX(id)");
        if (maxId == null) {
            txtflightid.setText("FO001");
        } else {
            long id = Long.parseLong(maxId.substring(2)) + 1;
            txtflightid.setText("FO" + String.format("%03d", id));}}
catch (ClassNotFoundException | SQLException ex) {
        Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
    } }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        String id = txtflightid.getText();
    String flightname = txtflightname.getText();
    String source = txtsource.getSelectedItem().toString().trim();
    String depart = txtdepart.getSelectedItem().toString().trim();
    DateFormat da = new SimpleDateFormat("yyyy-MM-dd");
    String date = da.format(txtdate.getDate());
    String deptime = txtdtime.getText();
    String arrtime = txtarrtime.getText();
    String flightcharge = txtflightcharge.getText();
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
             con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst = con.prepareStatement("insert into
flight(id,flightname,source,depart,date,deptime,arrtime,flightcharge)values(?,?,?,?,?,?,?,?)");
            pst.setString(1,id);
            pst.setString(2,flightname);
            pst.setString(3,source);
            pst.setString(4,depart);
            pst.setString(5,date);
            pst.setString(6,deptime);
            pst.setString(7,arrtime);
            pst.setString(8,flightcharge);
            pst.executeUpdate();
            JOptionPane.showMessageDialog(null,"Flight Created...");
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(addflight.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
            Logger.getLogger(addflight.class.getName()).log(Level.SEVERE, null, ex);}}
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        this.hide(); }
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
```

```
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JTextField txtarrtime;
    private com.toedter.calendar.JDateChooser txtdate;
    private javax.swing.JComboBox<String> txtdepart;
    private javax.swing.JTextField txtdtime;
    private javax.swing.JTextField txtflightcharge;
    private javax.swing.JLabel txtflightid;
    private javax.swing.JTextField txtflightname;
    private javax.swing.JComboBox<String> txtsource;
}
```

User Creation page code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
public class userCreation extends javax.swing.JInternalFrame {
public userCreation() {
        initComponents();
        autoID();}
    Connection con;
     PreparedStatement pst;
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
            String id = txtuserid.getText();
            String firstname = txtfirstname.getText();
            String lastname = txtlastname.getText();
            String username = txtusername.getText();
            String password = txtpassword.getText();
          try {
            Class.forName("com.mysql.cj.jdbc.Driver");
             con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
            pst = con.prepareStatement("insert into
user(id,firstname,lastname,username,password)values(?,?,?,?,?)");
            pst.setString(1,id);
            pst.setString(2,firstname);
            pst.setString(3,lastname);
            pst.setString(4,username);
            pst.setString(5,password);
            pst.executeUpdate();
```

```
        JOptionPane.showMessageDialog(null,"User Created...");
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(addflight.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(addflight.class.getName()).log(Level.SEVERE, null, ex);
    }}
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.hide();
}
public void autoID() {
try {Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3308/airline", "root", "");
    Statement s = con.createStatement();
    ResultSet rs = s.executeQuery("select MAX(id) from user");
    rs.next();
    String maxId = rs.getString("MAX(id)");
    if (maxId == null) {
        txtuserid.setText("UO001");
    } else {
        long id = Long.parseLong(maxId.substring(2)) + 1;
        txtuserid.setText("UO" + String.format("%03d", id));}
} catch (ClassNotFoundException | SQLException ex) {
    Logger.getLogger(addCustomer.class.getName()).log(Level.SEVERE, null, ex);
}}  private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JPanel jPanel1;
private javax.swing.JTextField txtfirstname;
private javax.swing.JTextField txtlastname;
private javax.swing.JPasswordField txtpassword;
private javax.swing.JLabel txtuserid;
private javax.swing.JTextField txtusername;}
```