



## **Report on Capstone Project**

19CSE204

Object Oriented Paradigm

### **Title of Project:**

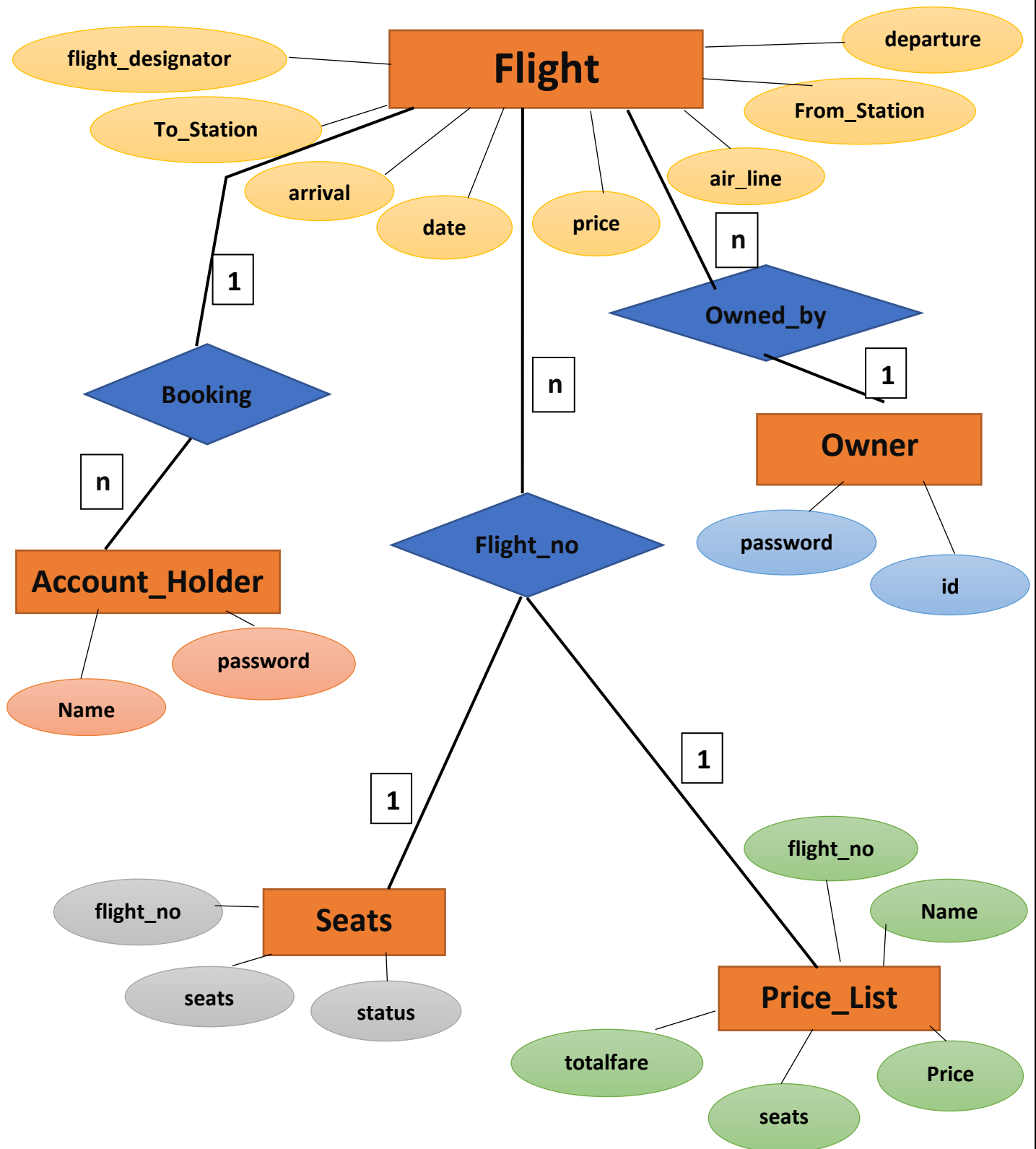
Airline Management System

## About Our application:

Vimana Aarakshanam is an airline management application which is based on booking the tickets by a customer. The application serves the services at both the customer end as well as owner end by using this application owner of the management can be able to access and manipulate the details of the flights he can add a flight ,he can add seats to flights, delete a flight if it's not running now and can be able to see the customers who are accessing the application As well parallelly it will allow the customer to sign in and login into application and he can search for flights as per his/her requirement so once he searches for trial he can book a flight once booking is done they can print the ticket



## ER-Diagram of Database:



## UML diagram for JAVA:



Adobe Acrobat  
PDFXML Document

*--Double click to open*

### Drivelink:

<https://drive.google.com/file/d/1k0kYSIhbtJdqjRRVTC1R7I9GG7KCXXR8/view?usp=sharing>

## **Application windows:**

**1) User login**

**2) User**

**3) Tracker info**

**4) Booking**

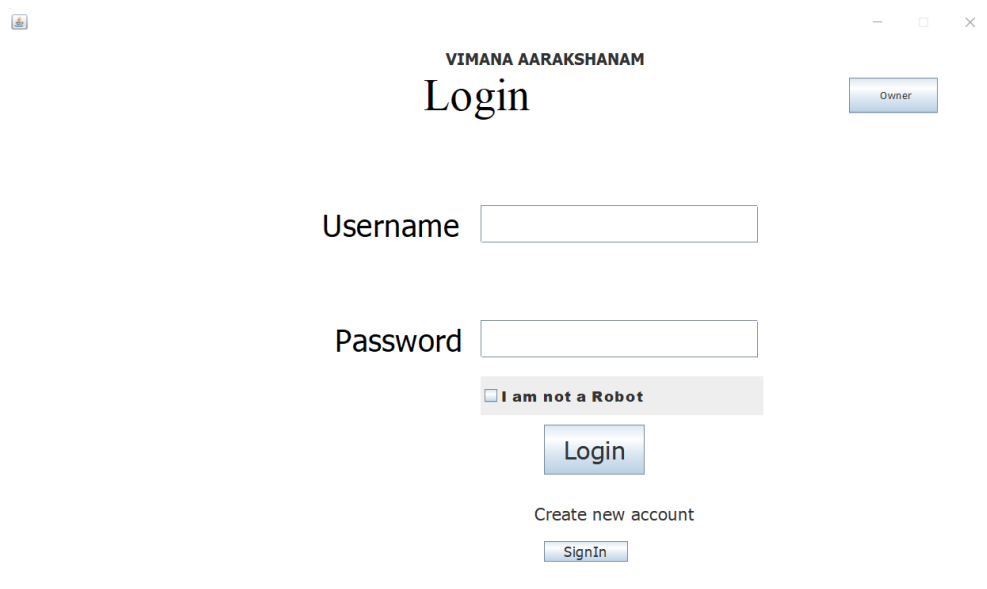
**5) Owner**

**6) Ownerinside**

1. Show flights
2. Add flight
3. Show account holders
4. Add Seats
5. Delete flight
6. Log out

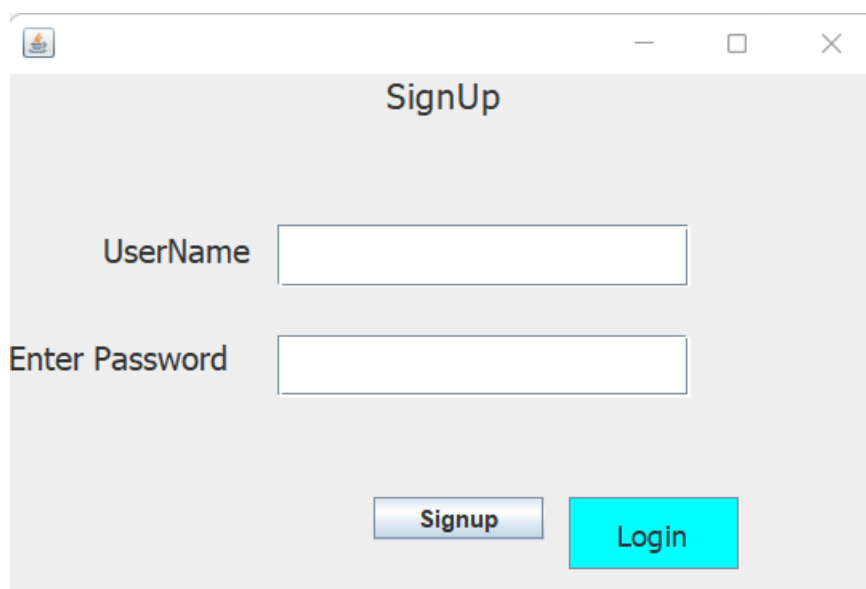
## USERLOGIN:

- The purpose of the window is user can login to the application by entering credential's [ username, password] if user already had account.



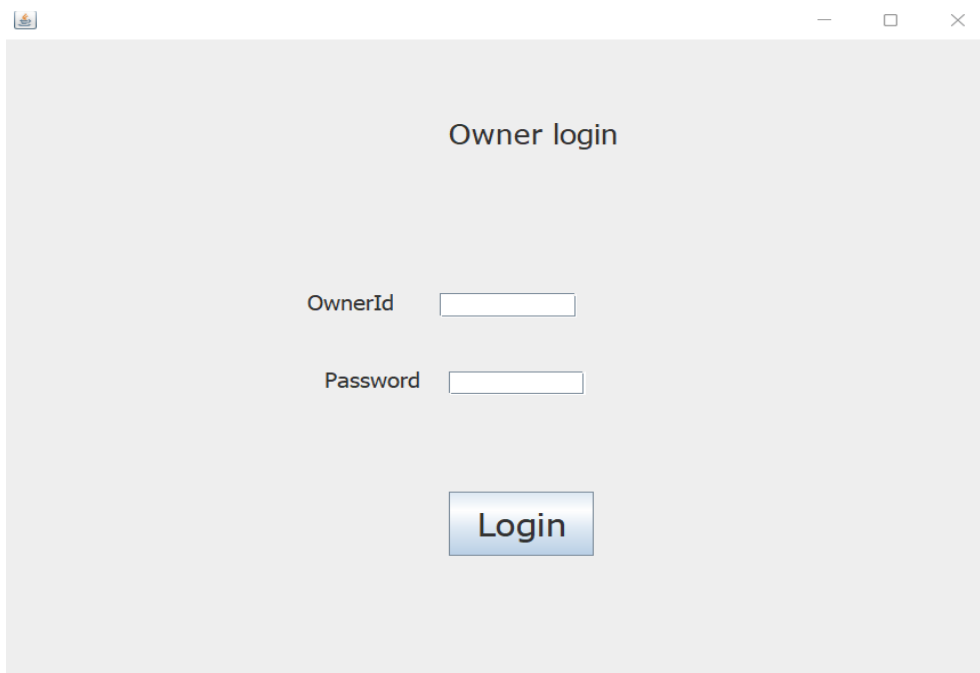
The screenshot shows a window titled "VIMANA AARAKSHANAM Login". In the top right corner, there is a button labeled "Owner". The main form contains a "Username" label followed by a text input field, a "Password" label followed by a text input field, and a checkbox labeled "I am not a Robot". Below these fields is a blue "Login" button. At the bottom of the form, there is a link "Create new account" and a blue "SignIn" button.

- If the user doesn't have account, down below there is a button to "sign-up". Then another window will get pop-up.



The screenshot shows a window titled "SignUp". It contains a "UserName" label followed by a text input field, and an "Enter Password" label followed by a text input field. At the bottom of the window, there are two buttons: a blue "Signup" button and a red "Login" button.

- In that window the user can enter his name and password, then user need to click “sign-up” button. then the account will be created with that credential’s entered by user.
- At the top right corner there is a “owner”. if the owner wants to enter into his portal, then, in that there is owner id and password.
- If the owner enters the details correctly then, “owner inside” window will be open then, then the owner can use the options in it.

A screenshot of a software window titled "Owner login". The window has a light gray background and standard window controls (minimize, maximize, close) in the top right corner. In the center, there are two input fields: the first is labeled "OwnerId" and the second is labeled "Password". Below these fields is a blue button with the text "Login" in white. The window is set against a white background.

Owner login

OwnerId

Password

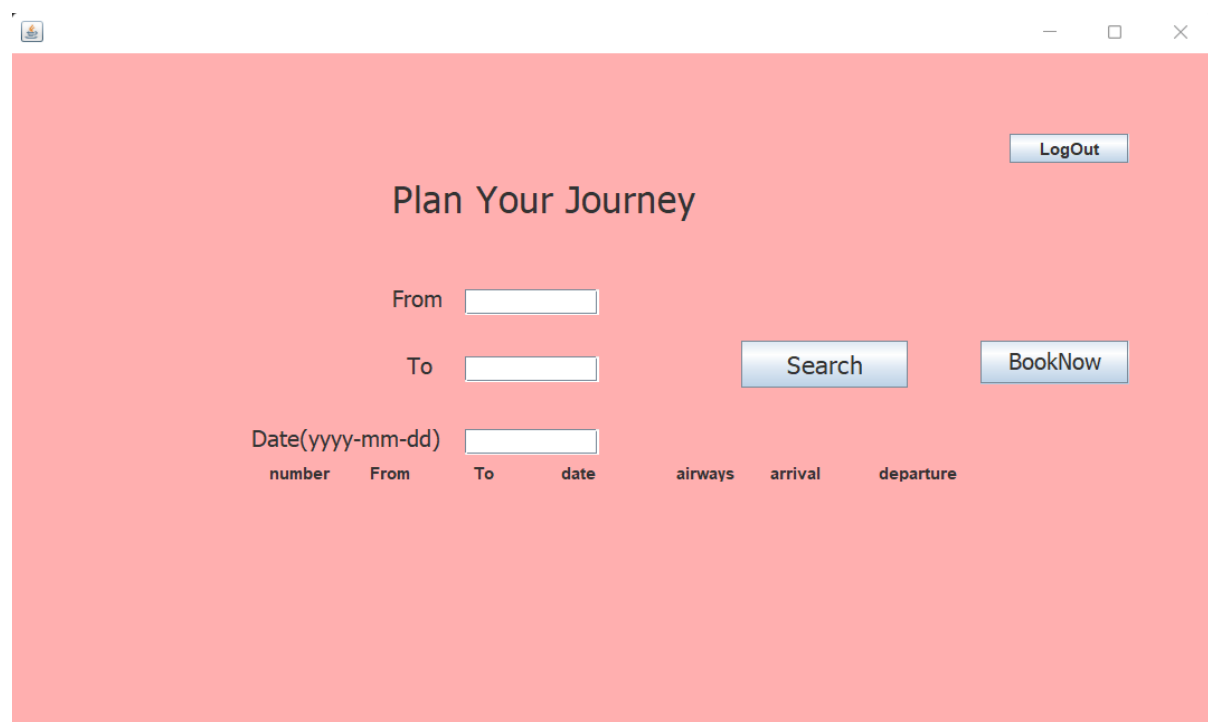
Login

## USER:

- In this window the user can see the flight details by entering the from, to and date of journey.
- Then the user can click search button to see the details of the flight which he needed at that time and place.
- If the flight exist then the **number, from, to, date, airways, arrival, departure** details will be displayed.
- If the user finds the correct flight, then there a button “Book Now”

To book flight, if the user clicks that button, then a new window will be open to book flight.

- And there a button to log out if the user wants to log out.



The screenshot shows a web application window with a light pink background. At the top right, there is a 'LogOut' button. The main heading is 'Plan Your Journey'. Below this, there are three input fields: 'From', 'To', and 'Date(yyyy-mm-dd)'. To the right of the 'To' field are two buttons: 'Search' and 'BookNow'. Below the input fields, there is a table with the following headers: 'number', 'From', 'To', 'date', 'airways', 'arrival', and 'departure'. The table body is currently empty.



## TICKET INFO:

**Passenger Name**

sai nithin

Name	Flightnumber	nuberofseats	price/ticket	totalprice
name	flight_no	seats	price	totalfare
0	0	0	0	0

**Flight number**

230

number	From_Sta...	To_Station	Date	airways	arrival	departure	price
230	Kolkata	Delhi	2021-12-22	Indigo	7:10 am	9:25 am	12000

- In this window the user can see the ticket details of his journey and flight details by giving flight number.
- If the passenger enters their name, then click “print” button then,  
The user can see the **name, flight number, seats, price, total fare** of the journey of that user.
- If the user enters flight number, then, they can see the  
**Number, from - To station, Date, Airways, Arrival, Departure, Price**  
Of the flight.

## BOOKING:

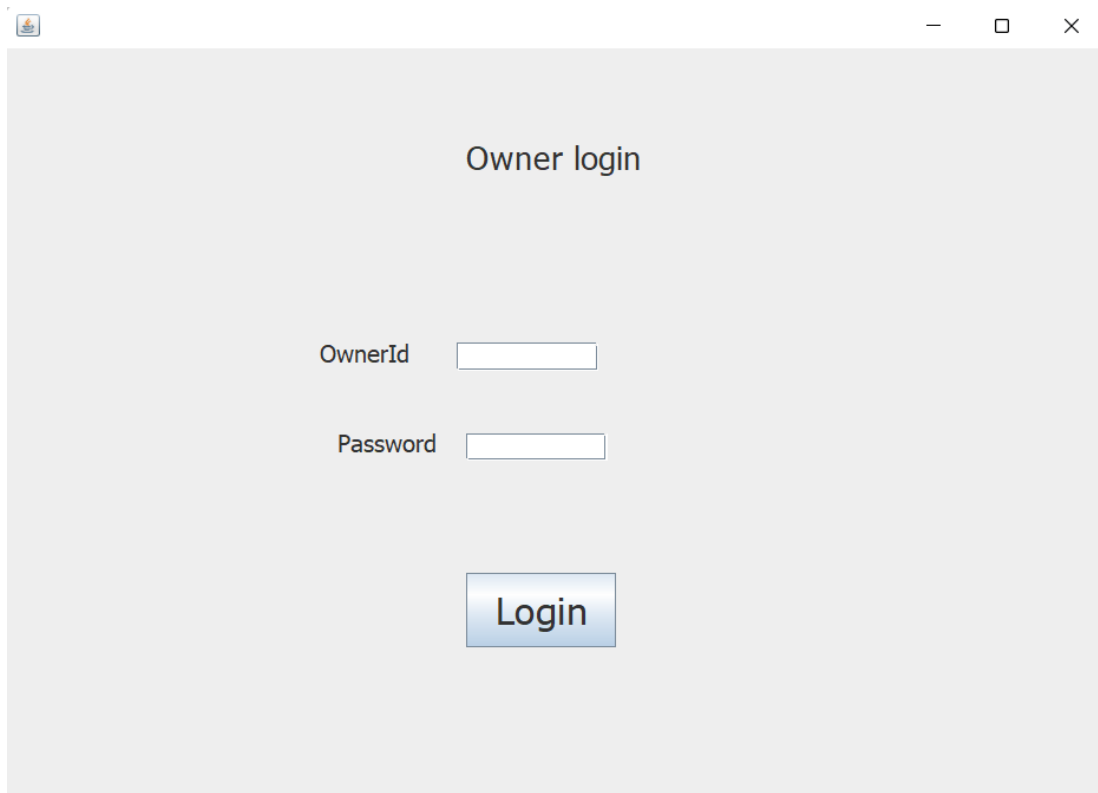
- Once searching for the train is done the customer can book a flight so for that first customer have to find the searched flight now customer can be able to see the flight details and the seats availability in the particular flight.
- The part of the book window will serve this this consists of text acceptor with flight number and a **Jbutton** labelled find flight once button is clicked user can see the details of the flight and seat availability and second part will have text acceptors as follows:
  - 1) Name
  - 2) Seat no
  - 3) No of seats
  - 4) Price
- And a other two **Jbuttons** labelled Calculate fare and book so once customer finalizes the ticket he can click on book now so once the button is clicked the seat in particular flight will get booked
- After ticket is booked customer can print the ticket by clicking the **Jbutton** labelled Print ticket

The screenshot displays a Java Swing window titled "BookNow". The window has a standard title bar with minimize, maximize, and close buttons. The main content area is light gray and contains the following elements:

- Title:** "BookNow" in a bold, black, serif font, centered at the top.
- Logout Button:** A rectangular button with a blue gradient and the text "Logout" in black, located in the top right corner.
- Flight Number:** A text label followed by a white text input field.
- Find F... Button:** A rectangular button with a blue gradient and the text "Find F..." in black, positioned below the Flight Number field.
- Form Fields:** Four text input fields arranged vertically on the left side, labeled "Name", "Seat no", "Number Of Seats", and "Price".
- Buttons:** Two rectangular buttons with blue gradients are located to the right of the form fields: "Caluclate F..." (note the typo) and "Book".
- Print Ti... Button:** A rectangular button with a blue gradient and the text "Print Ti..." in black, located at the bottom center of the window.
- Seat Availability:** A text label located on the right side of the window, next to a large, empty rectangular area.

## OWNER:

- ✚ An owner is the person who makes policy.
- ✚ Identify and access threats in the management.
- ✚ Inspect and monitor the whole database.
- ✚ And initiate necessary changes in the database.
- ❖ Owner need to login with his details first
  - Where the control panel created using **JPanel** contains
- ✓ Login page asking for the ownerId and Password mentioned with **JLabels**.
- ✓ And to enter details owner can enter details in the **JTextField** which enables the end user to enter his/ her login details.
- ✓ to enter details owner can enter details in the **JPasswordField** which check's whether the password entered by user are correct or not.
- ✓ At the end there is a **JButton** which enables owner to login/access to the database.



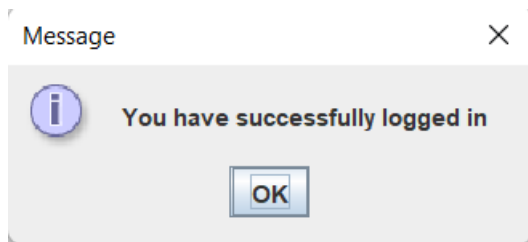
Owner login

OwnerId

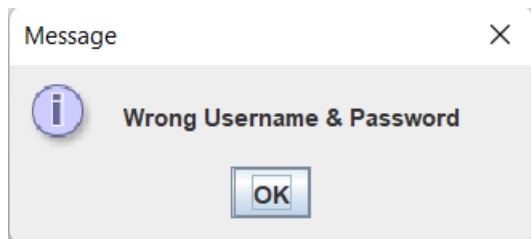
Password

Login

- ✓ If the details given by the user are correct it pop's a notification containing successfully logged in this is created using **JOptionPane**.



- ✓ And vice versa if invalid details entered saying invalid details.

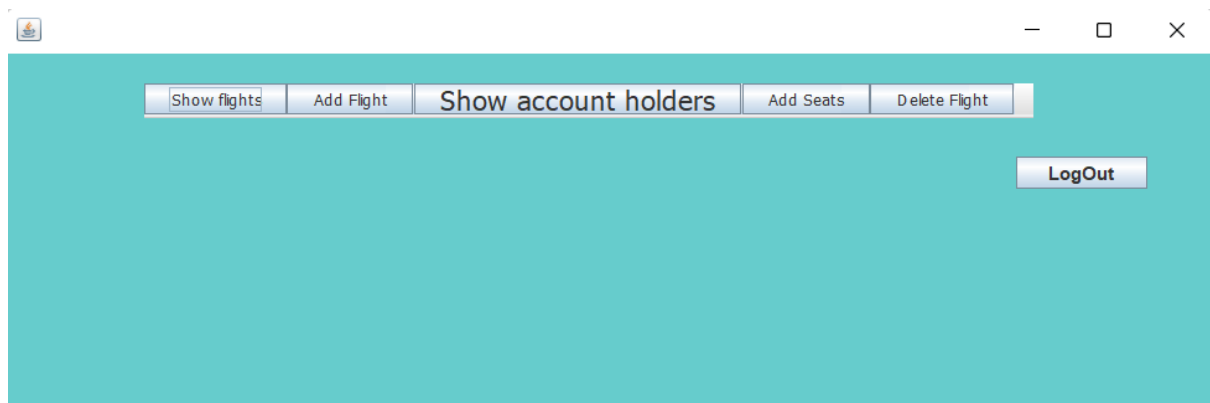


- ✓ After successfully logged in button click the ok button to access the ownerinside details.

## OWNERINSIDE:

❖ **Ownerinside** details includes the following details:

1. Show flights
2. Add flight
3. Show account holders
4. Add Seats
5. Delete flight
6. Log out



➤ This content panel contains the following details:

### 1) **Show flight:**

- After clicking the show flight button it displays the available flight's details.

### ✓ **Flight\_List:**

- The purpose of this window is to show all the flight details provided by our Airlines management system, added by the owner. This flight details window is only accessed by the owner.

Flight details table include:

1. Flight designator (flight number)
2. From station
3. To station
4. Date
5. Airline (Flight name)
6. Arrival
7. Departure
8. Price

This Content pane contains:

- Flights button which shows all the above-mentioned details once clicked, using **JButton** and **JLabel** showing “Flight Number”.
- It contains one back button which redirects to the owner window, using **JButton**.
- It also contains scrollbar which helps to see multiple details in the window itself, using **JScrollBar**.
- Contains table consisting of all flight details, using **Jtable**.
- The Default Table Model is connected to prepared statement (dbms end that is to retrieve table info), to execute query through **Action events**.
- **Action Listener** used to close or dispose the content pane.
- Includes **SQL exception** to print stack trace.



number	From_Station	To_Station	Date	airways	arrival	departure	price
230	Kolkata	Delhi	2021-12-22	Indigo	7:10 am	9:25 am	12000
845	Hyderabad	Nagpur	2021-12-11	Air India	6 pm	6:55 pm	11000
1010	Srinagar	Delhi	2021-12-20	JetLite	2:30 pm	3:30 pm	10000
1110	Kolkata	Guwahati	2021-12-21	Alliance Air	5:50 am	7:25 am	10000
1111	nagpur	kasmhir	2021-12-16	Air india	9am	12am	2000
1234	efnjv	foi	2020-10-10	edrf	9	10	10000
1235	bapatla	chirala	2020-10-10	garuda	9	9.10	
1305	Dhaka	Kolkata	2021-12-13	SpiceJet	9:20 am	9:40 am	
1780	Chennai	Mumbai	2021-11-09	Spice Jet	6:50 am	8:45 am	
1818	Dubai	Mumbai	2021-12-27	Indigo	6 pm	10:30 pm	
1847	Kolkata	Dhaka	2021-12-30	SpiceJet	7:05 am	8:35 am	
1971	Bangalore	Singapore	2021-12-05	Indigo	9:20 pm	4:30 pm	
2400	Bangalore	Mumbai	2021-11-19	Indigo	10 pm	11:30 pm	
2540	Mumbai	Bangalore	2021-11-19	Indigo	2 pm	3:30 pm	
2744	Mumbai	Dubai	2021-12-27	Indigo	1:10 pm	2:55 pm	
2810	Guwahati	Kolkata	2021-12-15	Alliance Air	10:25 am	11:55 am	
3010	Srinagar	Delhi	2021-12-20	Airasia India	8:02 am	9:02 am	
3679	Nagpur	Hyderabad	2021-12-12	Air India	10 am	11 am	
3692	Vizag	Delhi	2021-12-18	Airasia	9:35 am	12 am	
3780	Chennai	Mumbai	2021-11-29	Air India	6:50 am	8:45 am	
3877	Bangkok	Delhi	2021-11-28	SpiceJet	3:50 am	6:25 am	

## 2) Add flight:

- A owner need to add a flight so that the customers can book them so that purpose will be served by the add\_flight window if a flight is to be reflected at the customer end before that owner of the management should add the flight so that whenever customer searches for a particular journey he/she will be able to see result for their search.so in this window when a owner logs in with his id and password he can add a flight on clicking on add flight button in menu\_bar onclick it will display a window which accepts the following details:

1. Flight Number
2. From
3. To
4. Date of journey
5. Airline name
6. Arrival
7. Departure

## 8. Price

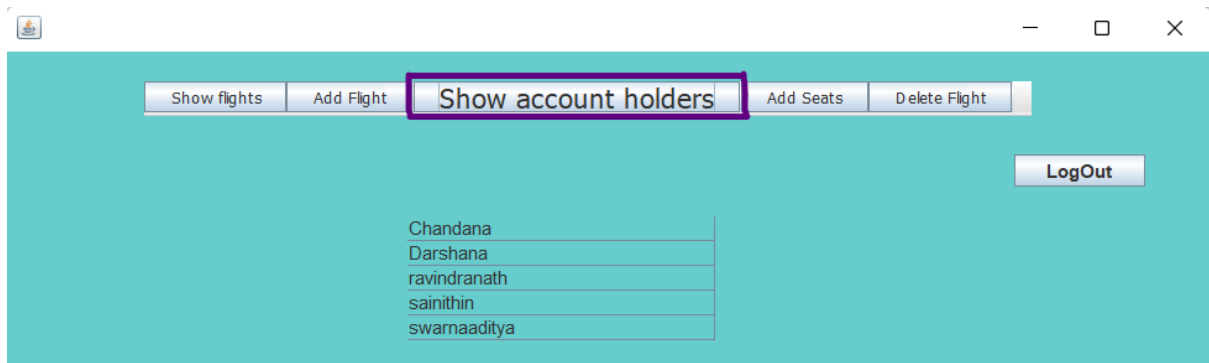
- And the window consist a **Jbutton** labelled Add flight
- Once it is clicked then the entered details will be added to already existing table consisting of list of flights named “flight” and a back button which will return to owner home page.

The image displays two screenshots of a Java Swing application window. The top screenshot shows the main menu bar with buttons: "Show flights", "Add Flight" (highlighted with a red box), "Show account holders", "Add Seats", "Delete Flight", and "LogOut". The bottom screenshot shows the "Add Flight" form with input fields for "Flight Number", "From", "To", "Date(yyyy-mm-dd)", "Air line", "Arrival", "Departure", and "Price", along with a back button and an "Add Fli..." button.

### 3) Show account holders:

- This displays the list of account holders from the back-end data base using created using **JMenuBar**.



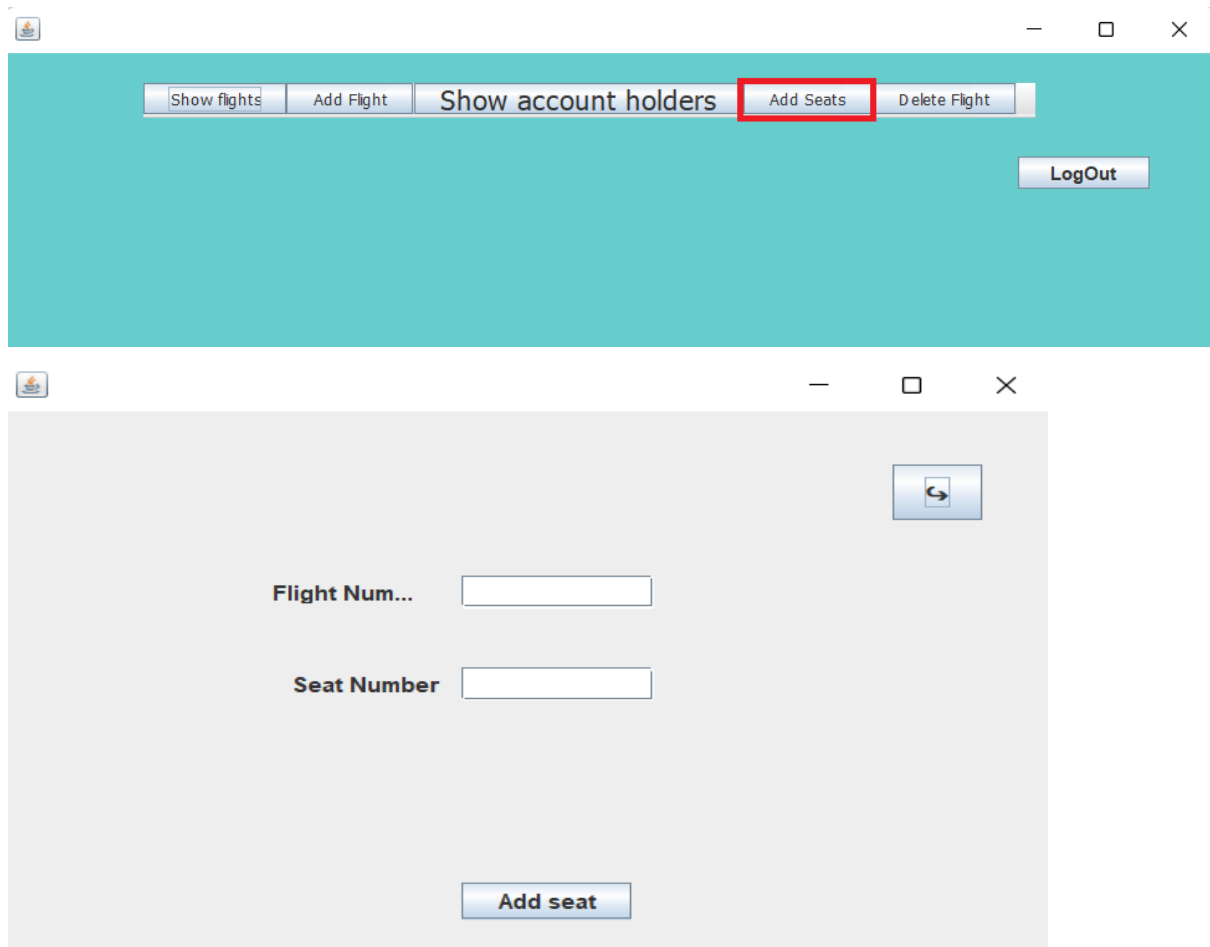


#### 4) Add seats:

- A owner need to add seats to every flight so that the customers can book them so that purpose will be served by the add\_seats window if a seats in a flight is to be reflected at the customer end before that owner of the management should add seats to the flight so that whenever customer searches for a particular journey and tries to book a seat he/she will be able to see result for their search and can find the status of seats in particular flight.so in this window when a owner logs in with his id and password he can add a seats to flight on clicking on add seats button in menu\_bar onclick it will display a window which accepts the following details:

1. Flight Number
2. Seat number

- And the window consists of a **Jbutton** labelled Add seat
- Once it is clicked then the entered details will be added to already existing table consisting of list of flights named “seats” the default status of any seat will be not booked and a back button which will return to owner home page.



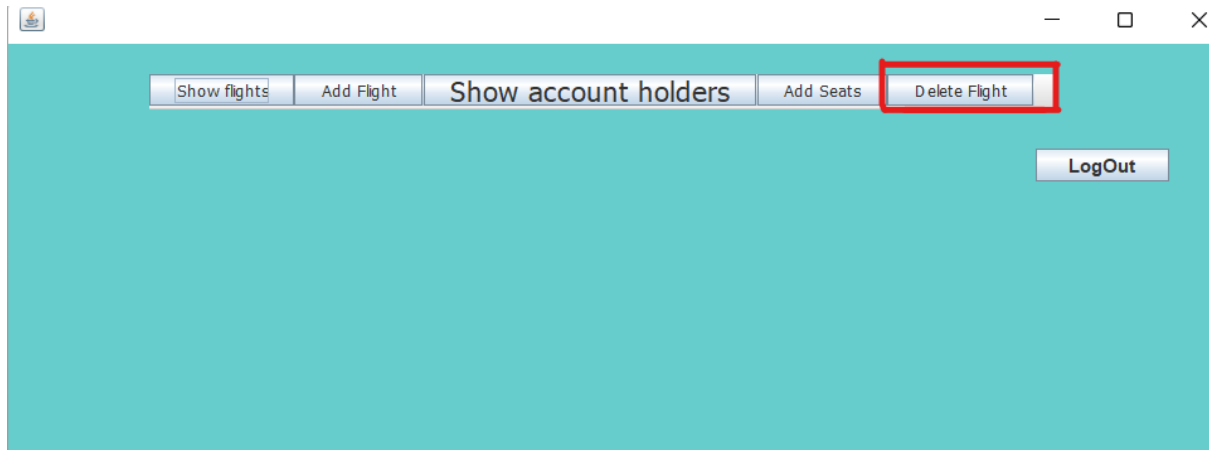
## 5) Delete flight:

The purpose of this window is to delete the unwanted flights, repairing flights from the owner end. This window is only accessed by owner.

The content pane consists of:

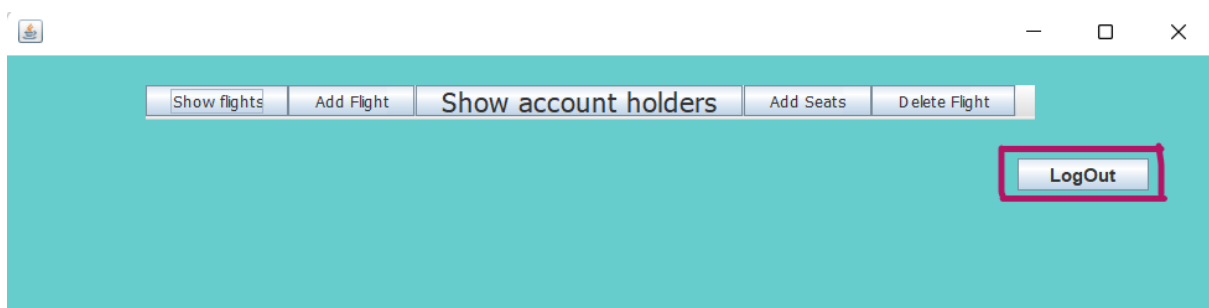
- **Text field** to enter the flight designator or number (to which owner wants to delete) through **Action Listener**.
- Button named Delete to complete the action and back button to redirect to owner page using **JButton**.
- The Default Table Model is connected to prepared statement (dbms end, that is to delete the data from flight\_details as required), to execute query through **Action events** and keeps on updating as per the changes done.

- After deleting, a message dialog box will pop up signifying “Flight deleted successfully” using **JOptionPane**.
- If the entered flight number is wrong or doesn’t exist then a message dialog box will appear signifying “No flight found” using **JOptionPane**.
- Includes **SQL exception** to print stack trace.



## 6) Log out:

- **Action Listener** used to log out the content pane.
- Includes **SQL exception** to print stack trace.
- And if the owner logout’s from the panel the page redirects to the main Login page i.e., user login page.
- This logout button is created using **JButton**.



**Sample Code (user login page):**

```
package connector;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
import javax.swing.border.EmptyBorder;

public class UserLogin extends JFrame {
```

```
private static final long serialVersionUID = 1 ;

public static JTextField textField;

private JPasswordField passwordField;

private JButton btnNewButton;

private JLabel label;

private JPanel contentPane;

/**

 * Launch the application.

 */

public static void main(String[] args) {

    EventQueue.invokeLater(new Runnable() {

        public void run() {

            try {

                UserLogin frame = new UserLogin();

                frame.setVisible(true);

            } catch (Exception e) {

                e.printStackTrace();

            }

        }

    });

}

/**

 * Create the frame.

 */

public UserLogin() {

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    setBounds(450, 190, 1014, 597);
```

```
setResizable(false);  
contentPane = new JPanel();  
contentPane.setBackground(Color.pink);  
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  
setContentPane(contentPane);  
contentPane.setLayout(null);
```

```
JLabel lblNewLabel = new JLabel("Login");  
lblNewLabel.setForeground(Color.BLACK);  
lblNewLabel.setFont(new Font("Times New Roman", Font.PLAIN, 46));  
lblNewLabel.setBounds(423, 13, 273, 93);  
contentPane.add(lblNewLabel);
```

```
textField = new JTextField();  
textField.setFont(new Font("Tahoma", Font.PLAIN, 32));  
textField.setBounds(481, 170, 281, 39);  
contentPane.add(textField);  
textField.setColumns(10);
```

```
passwordField = new JPasswordField();  
passwordField.setFont(new Font("Tahoma", Font.PLAIN, 32));  
passwordField.setBounds(481, 286, 281, 39);  
contentPane.add(passwordField);
```

```
JLabel lblUsername = new JLabel("Username");  
lblUsername.setBackground(Color.BLACK);  
lblUsername.setForeground(Color.BLACK);
```

```
lblUsername.setFont(new Font("Tahoma", Font.PLAIN, 31));  
lblUsername.setBounds(321, 178, 150, 25);  
contentPane.add(lblUsername);
```

```
JLabel lblPassword = new JLabel("Password");  
lblPassword.setForeground(Color.BLACK);  
lblPassword.setBackground(Color.CYAN);  
lblPassword.setFont(new Font("Tahoma", Font.PLAIN, 31));  
lblPassword.setBounds(334, 280, 137, 52);  
contentPane.add(lblPassword);
```

```
JCheckBox checkBox1 = new JCheckBox("I am not a Robot");  
checkBox1.setFont(new Font("Arial Black", Font.BOLD, 15));  
checkBox1.setBounds(481,343,286,39);  
contentPane.add(checkBox1);
```

```
btnNewButton = new JButton("Login");  
btnNewButton.setFont(new Font("Tahoma", Font.PLAIN, 26));  
btnNewButton.setBounds(545, 392, 102, 50);  
btnNewButton.addActionListener(new ActionListener() {  
  
    public void actionPerformed(ActionEvent e) {  
        String userName = textField.getText();  
        @SuppressWarnings("deprecation")  
        String password = passwordField.getText();  
        try {
```

```

        Connection connection = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/example",
        "root", "Aditya03!");

```

```

        PreparedStatement st = (PreparedStatement) connection
        .prepareStatement("Select name, password from account_holders where
name=? and password=?");

```

```

        st.setString(1, userName);
        st.setString(2, password);
        ResultSet rs = st.executeQuery();
        if (rs.next()) {
            dispose();
            UserLogin ah = new UserLogin ();
            ah.setTitle("Welcome");
            ah.setVisible(true);
            JOptionPane.showMessageDialog(btnNewButton, "You have successfully
logged in");
            ah.dispose();
            use frame = new use();
            frame.setVisible(true);
        } else {

            JOptionPane.showMessageDialog(btnNewButton, "Wrong Username &
Password");

        }
    } catch (SQLException sqlException) {
        sqlException.printStackTrace();
    }
}

```



```
});
```

```
contentPane.add(btnNewButton);
```

```
label = new JLabel("");
```

```
label.setBounds(0, 0, 1008, 562);
```

```
contentPane.add(label);
```

```
JLabel lblNewLabel_1 = new JLabel("");
```

```
lblNewLabel_1.setIcon(new ImageIcon("D:\\Desktop\\Projectimages\\logo (3).png"));
```

```
lblNewLabel_1.setBounds(67, 133, 204, 188);
```

```
contentPane.add(lblNewLabel_1);
```

```
JLabel lblNewLabel_2 = new JLabel("VIMANA AARAKSHANAM ");
```

```
lblNewLabel_2.setFont(new Font("Tahoma", Font.BOLD, 16));
```

```
lblNewLabel_2.setBounds(446, 10, 232, 25);
```

```
contentPane.add(lblNewLabel_2);
```

```
JLabel lblNewLabel_3 = new JLabel("Create new account");
```

```
lblNewLabel_3.setFont(new Font("Tahoma", Font.PLAIN, 18));
```

```
lblNewLabel_3.setBounds(535, 469, 303, 25);
```

```
contentPane.add(lblNewLabel_3);
```

```
JButton btnNewButton_1 = new JButton("SignIn");
```

```
btnNewButton_1.setFont(new Font("Tahoma", Font.PLAIN, 15));
```

```
btnNewButton_1.addActionListener(new ActionListener() {
```

```
    public void actionPerformed(ActionEvent e) {
```

```
        Project1 p2=new Project1();
        p2.setVisible(true);
        p2.setLayout(null);
    }
});
btnNewButton_1.setBounds(545, 509, 85, 21);
contentPane.add(btnNewButton_1);

JButton btnNewButton_2 = new JButton(" Owner");
btnNewButton_2.setFont(new Font("Tahoma", Font.PLAIN, 11));
btnNewButton_2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        dispose();
        owner frame = new owner();
        frame.setVisible(true);
    }
});
btnNewButton_2.setBounds(853, 41, 90, 36);
contentPane.add(btnNewButton_2);

}

}

class Project1
{
    JFrame f1=new JFrame();
```

```
Project1()
```

```
{
```

```
f1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
f1.setVisible(true);
```

```
f1.setBounds(100,100,450,300);
```

```
JLabel lblNewLabel = new JLabel("SignUp");
```

```
lblNewLabel.setFont(new Font("Tahoma", Font.PLAIN, 18));
```

```
lblNewLabel.setHorizontalAlignment(SwingConstants.CENTER);
```

```
f1.getContentPane().add(lblNewLabel, BorderLayout.NORTH);
```

```
JTextField textField = new JTextField();
```

```
textField.setBounds(135, 76, 206, 31);
```

```
f1.add(textField);
```

```
textField.setColumns(1);
```

```
JLabel lblNewLabel_1 = new JLabel("UserName");
```

```
lblNewLabel_1.setBounds(446, 10, 232, 25);
```

```
lblNewLabel_1.setFont(new Font("Tahoma", Font.PLAIN, 16));
```

```
lblNewLabel_1.setBounds(47, 71, 82, 36);
```

```
f1.add(lblNewLabel_1);
```

```
JTextField PasswordField_1 = new JPasswordField();
```

```
PasswordField_1.setBounds(135, 131, 206, 31);

f1.add(PasswordField_1);

PasswordField_1.setColumns(10);


JLabel lblPassword = new JLabel("Enter Password");

lblPassword.setFont(new Font("Tahoma", Font.PLAIN, 16));

lblPassword.setBounds(534, 280, 88, 36);

f1.add(lblPassword);


JButton btnNewButton_1 = new JButton("Login");

btnNewButton_1.setBackground(Color.CYAN);

btnNewButton_1.setFont(new Font("Tahoma", Font.PLAIN, 15));

btnNewButton_1.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        f1.dispose();

        UserLogin p2=new UserLogin();

        p2.setVisible(true);

        p2.setLayout(null);

    }

});

f1.setVisible(true);

f1.setLayout(null);

btnNewButton_1.setBounds(280,212,85,36);

f1.add(btnNewButton_1);


JButton btnNewButton = new JButton("Signup");

btnNewButton.setBounds(183, 212, 85, 21);
```

```
f1.setVisible(true);

f1.setLayout(null);

f1.add(btnNewButton);

btnNewButton.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {

        String userName = textField.getText();

        String password = PasswordField_1.getText();

        if(textField.getText()!=null)

            try {

                Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/example",

                    "root", "Aditya03!");

                PreparedStatement sql = (PreparedStatement) connection

                    .prepareStatement("INSERT INTO account_holders VALUES(?,?) ");

                sql.setString(1,userName);

                sql.setString(2,password);

                sql.executeUpdate();

                JOptionPane.showMessageDialog(btnNewButton, "You have successfully Signed in now

you can login ");

            } catch (SQLException sqlException) {
```

```
JOptionPane.showMessageDialog(btnNewButton, "user name already exists");  
    }  
}  
});
```

```
}
```

```
protected void setLayout(Object object) {  
    // TODO Auto-generated method stub  
  
}
```

```
protected void setVisible(boolean b) {  
    // TODO Auto-generated method stub  
  
}
```

```
protected void setTitle(String string) {  
    // TODO Auto-generated method stub  
  
}  
}
```

## DATA BASE MANAGEMENT:

### Entities in database:-

1. Flight
2. Account\_holders
3. Owners
4. Flights
5. Seats

➔ Flight entity consists all the data related to flight which can be accessible by customer so that he can search for a flight

### Attributes in flight entity:

- Flight designator
- From station
- To station
- Date
- Airline
- Arrival
- Departure
- Price

Flight	Datatypes
Flight designator	Int (PRI)
From station	Char (10)
To station	Char (20)
Date	Date
Airline	Char (20)
Arrival	Char (20)
Departure	Char (20)
Price	int

➔Account\_holders entity holds the details of all the account holders who can be able to enter into application and search ,book a flight and owner can see the customers holds a account .

#### Attributes in Account\_holders:

- name
- Password

Account holder	Datatypes
name	Char (20) (PRI)
Password	Char (20)

➔Owner entity holds details of the owner

#### Attributes in Owner:

- ID
- Password

Owner	Datatypes
ID	Int
Password	Char (20)

➔Seats entity consists of details of seats availability of flights

#### Attributes in Seats:

- Flight number
- Status
- Seats

Seats	Datatypes
Flight number	Int (MUL)
Status	Char (20)
Seats	Int

➔Pricelist entity holds details of all the customers booked ticket

#### Attributes in Pricelist:

- Flight number
- name
- Seats
- Price
- Total fare

Price list	Datatypes
Flight number	Int (MUL)
name	Char (20)
Seats	Int
Price	Int
Total fare	Int



## DISPLAYING TABLES:

```
mysql> desc account_holders;
```

Field	Type	Null	Key	Default	Extra
name	char(20)	NO	PRI	NULL	
password	char(20)	NO		NULL	

2 rows in set (0.06 sec)

```
mysql> desc flight;
```

Field	Type	Null	Key	Default	Extra
flight_designator	int	NO	PRI	NULL	
From_Station	char(10)	YES		NULL	
To_Station	char(20)	YES		NULL	
date	date	YES		NULL	
air_line	char(20)	YES		NULL	
arrival	char(20)	YES		NULL	
departure	char(20)	YES		NULL	
price	int	YES		NULL	

8 rows in set (0.00 sec)

```
mysql> desc owner;
```

Field	Type	Null	Key	Default	Extra
id	int	YES		NULL	
password	char(20)	YES		NULL	

2 rows in set (0.00 sec)

```
mysql> desc pricelist;
```

Field	Type	Null	Key	Default	Extra
name	char(20)	YES		NULL	
flight_no	int	YES	MUL	NULL	
seats	int	YES		NULL	
price	int	YES		NULL	
totalfare	int	YES		NULL	

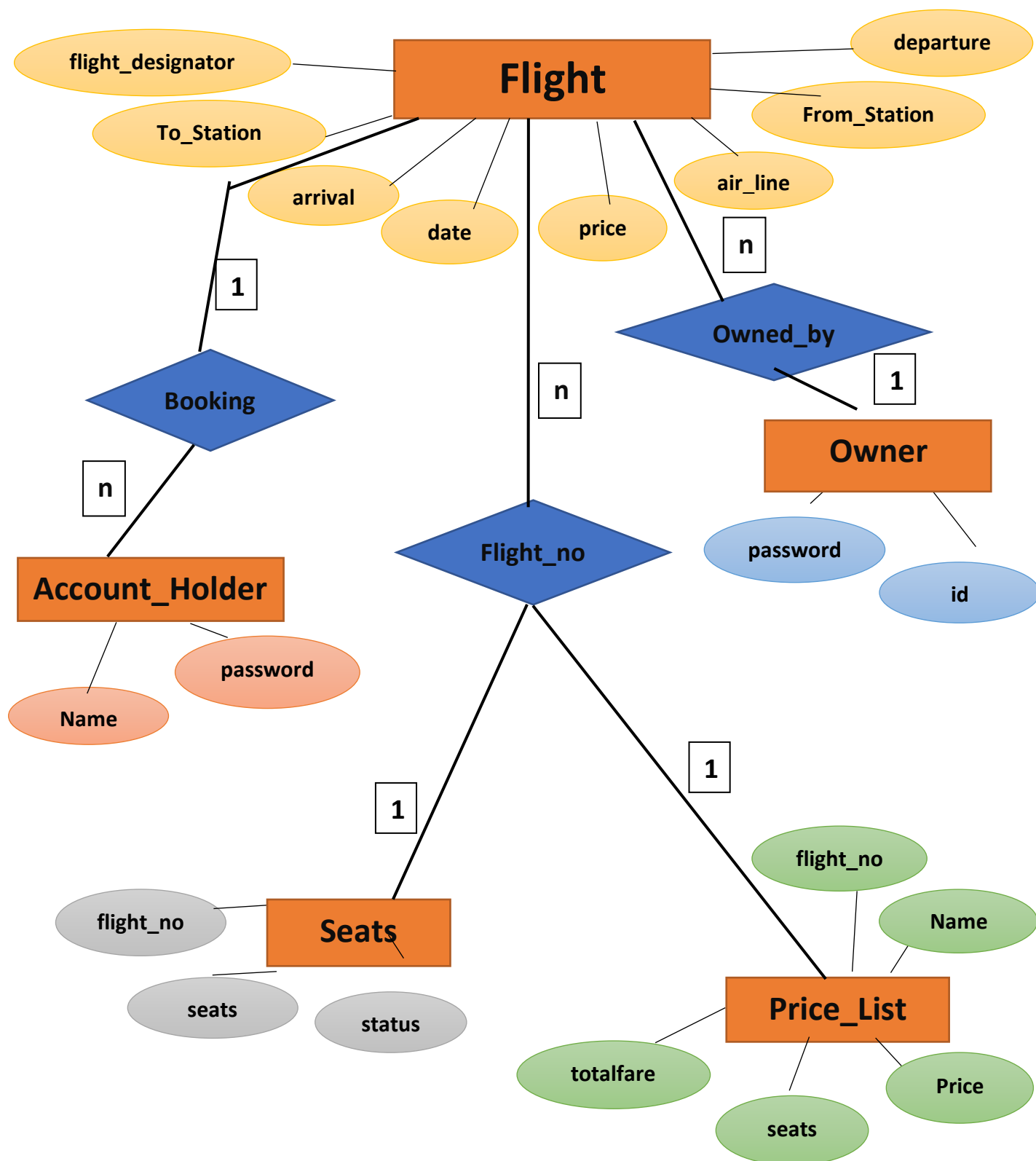
5 rows in set (0.00 sec)

```
mysql> desc seats;
```

Field	Type	Null	Key	Default	Extra
flight_no	int	YES	MUL	NULL	
seats	int	NO		NULL	
status	char(20)	YES		NULL	

3 rows in set (0.00 sec)

## E-R Diagram:



## TABLES:

```
mysql> select*from flight;
```

flight_designator	From_Station	To_Station	date	air_line	arrival	departure	price
230	Kolkata	Delhi	2021-12-22	Indigo	7:10 am	9:25 am	12000
845	Hyderabad	Nagpur	2021-12-11	Air India	6 pm	6:55 pm	11000
1010	Srinagar	Delhi	2021-12-20	JetLite	2:30 pm	3:30 pm	10000
1110	Kolkata	Guwahati	2021-12-21	Alliance Air	5:50 am	7:25 am	10000
1235	bapatla	chirala	2020-10-10	garuda	9	9.10	NULL
1305	Dhaka	Kolkata	2021-12-13	SpiceJet	9:20 am	9:40 am	NULL
1780	Chennai	Mumbai	2021-11-09	Spice Jet	6:50 am	8:45 am	NULL
1818	Dubai	Mumbai	2021-12-27	Indigo	6 pm	10:30 pm	NULL
1847	Kolkata	Dhaka	2021-12-30	SpiceJet	7:05 am	8:35 am	NULL
1971	Bangalore	Singapore	2021-12-05	Indigo	9:20 pm	4:30 pm	NULL
2400	Bangalore	Mumbai	2021-11-19	Indigo	10 pm	11:30 pm	NULL
2540	Mumbai	Bangalore	2021-11-19	Indigo	2 pm	3:30 pm	NULL
2744	Mumbai	Dubai	2021-12-27	Indigo	1:10 pm	2:55 pm	NULL
2810	Guwahati	Kolkata	2021-12-15	Alliance Air	10:25 am	11:55 am	NULL
3010	Srinagar	Delhi	2021-12-20	Airasia India	8:02 am	9:02 am	NULL
3679	Nagpur	Hyderabad	2021-12-12	Air India	10 am	11 am	NULL
3692	Vizag	Delhi	2021-12-18	Airasia	9:35 am	12 am	NULL
3780	Chennai	Mumbai	2021-11-29	Air India	6:50 am	8:45 am	NULL
3877	Bangkok	Delhi	2021-11-28	SpiceJet	3:50 am	6:25 am	NULL
4560	Bangalore	Kolkata	2021-11-30	Airasia	3:40 pm	7:40 pm	NULL
4605	Goa	Bangalore	2021-12-10	Air India	10 pm	11:15 pm	NULL
5004	Maharastra	Delhi	2021-12-01	JetSpace	9am	12am	13000
5208	Delhi	Bangkok	2021-11-28	SpiceJet	9:20 pm	3:00 am	NULL
5930	Bangalore	Delhi	2021-12-02	Indigo	9:55 am	12:35 pm	NULL
6543	Bapatla	Vijayawada	2021-12-01	Garuda	9am	10am	10000
7902	Vizag	Pune	2021-12-18	Airasia	4 am	5:30 am	NULL
8720	Cochin	Bangalore	2021-11-25	Alliance Air	7:10 pm	8:30 am	NULL
8899	Goa	Hyderabad	2021-12-01	Air India	1:30 pm	2:45 pm	NULL
9200	Hyderabad	Bangalore	2021-12-31	Airasia	9:25 am	11 am	NULL
9310	Jaipur	Delhi	2021-12-17	Alliance Air	8:30 pm	9:30 pm	NULL
9910	Singapore	Bangalore	2021-12-05	Indigo	5:30 am	8 am	NULL
11380	Hyderabad	Chicago	2022-01-31	Air India	8:55 pm	7:25 am	NULL
12345	bapatla	hyderabad	2020-10-10	garuda	9am	12am	NULL
38390	Paris	Chennai	2022-03-05	Kuwait Airways	1:10 pm	2:15 am	NULL
59296	Delhi	SanFrancisco	2022-06-21	Emirates	10:35 pm	2:15 pm	NULL
59321	Delhi	Los Angeles	2022-01-02	Atlantic	3:55 am	3:25 pm	NULL
72616	Chennai	Sydney	2022-04-29	Thai Airways	1:30 am	9:20 pm	NULL
72687	Nagpur	New York	2022-09-19	Air India	7:55 am	5:25 pm	NULL

38 rows in set (0.00 sec)

```
mysql> select*from seats;
```

flight_no	seats	status
2744	1	B
2744	2	NB
2744	3	NB
2744	4	NB
2744	5	NB
2744	6	NB
2744	7	NB
2744	8	NB
2744	9	NB
2744	10	NB
1110	1	NB
230	1	B
6543	1	NB
6543	2	B
6543	3	NB
5004	1	NB
5004	2	NB
5004	3	B
5004	4	NB
5004	5	NB

```
20 rows in set (0.04 sec)
```

```
mysql> select*from pricelist;
```

name	flight_no	seats	price	totalfare
Aditya	230	1	12000	12000
Kartheek	6543	1	10000	10000
Aditya	5004	1	13000	13000

```
3 rows in set (0.00 sec)
```

```
mysql> select*from owner;
```

id	password
123456	aditya
234567	Chandu

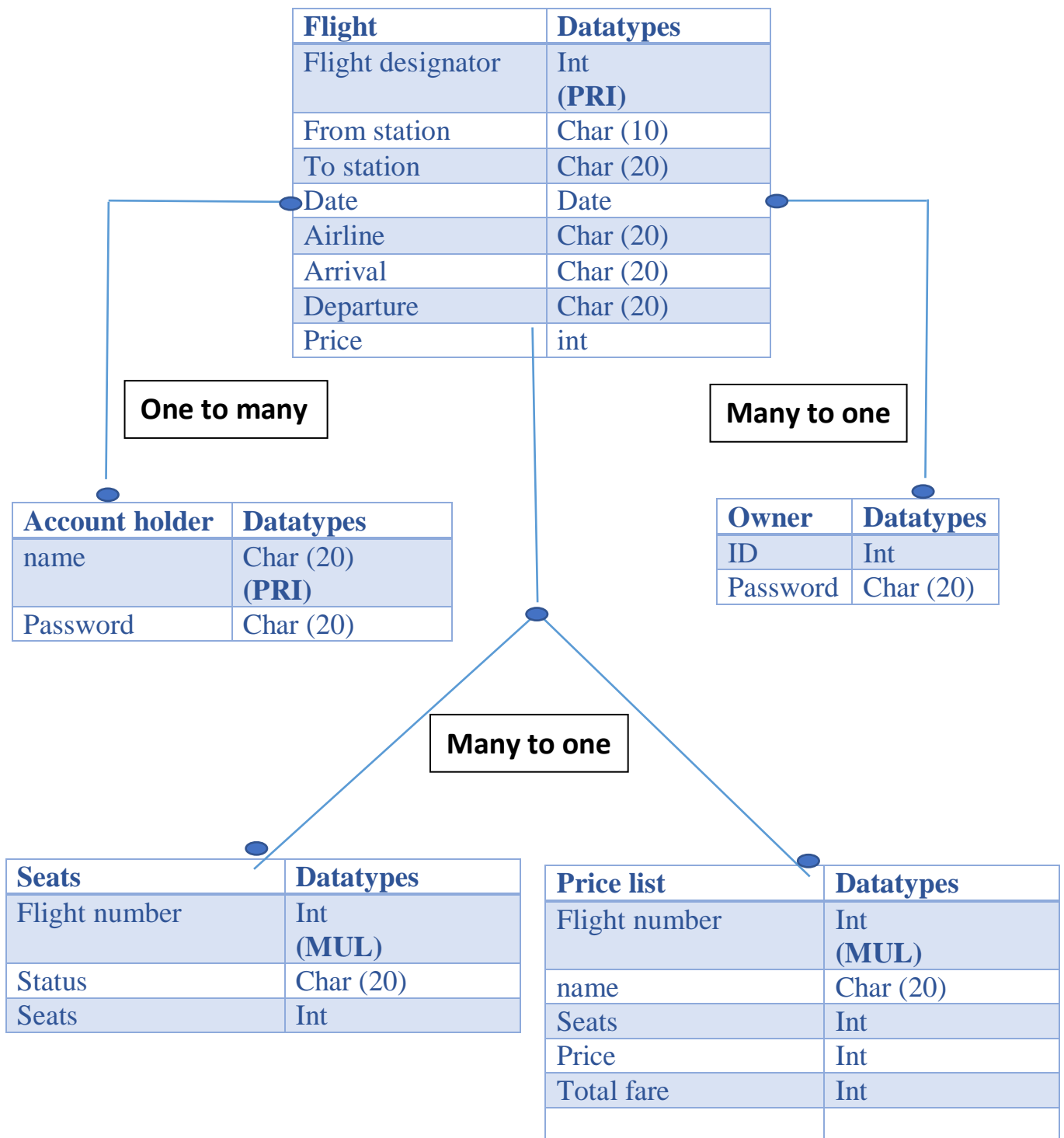
```
2 rows in set (0.00 sec)
```

```
mysql> select*from account_holders;
```

name	password
Chandana	chandu
Darshana	darshu
ravindranath	ravi
sainithin	sai
swarnaaditya	aditya

```
5 rows in set (0.01 sec)
```

## Relations:



**Learning Outcomes:**

After working on this project:

- develop plans with relevant people to achieve the project's goals
- break work down into tasks and determine handover procedures
- identify links and dependencies, and schedule to achieve deliverables
- estimate and cost the human and physical resources required, and make plans to obtain the necessary resources
- allocate roles with clear lines of responsibility and accountability.

**Acknowledgement:**

We would like to express my deepest appreciation to all those who provided me the possibility to complete this project. A special gratitude we give to our college, management, and Dr. B. S. Vidhyasagar & Ms. Karpagam mam, whose contribution in stimulating suggestions and encouragement, helped me to coordinate my project. We must appreciate the guidance given by other supervisor as well as the panels especially in our project presentation that has improved our presentation skills thanks to their comment and advice.