

## ACKNOWLEDGEMENT

The successful presentation of the **DBMS MINI PROJECT** would be incomplete without the mention of the people who made it possible and whose constant guidance crowned my effort with success.

We would like to extend my gratitude to the **MANAGEMENT, KAMMAVARI SANGHAM**, Bengaluru, for providing all the facilities to present the Database Application Mini Project.

We would like to extend my gratitude to **Dr. K.RAMA NARASIMHA**, Principal / Director, K. S. School of Engineering and Management, Bengaluru, for facilitating me to present the Database Application Mini Project.

We thank **Dr. K Venkata Rao**, Professor and Head, Department of Computer Science and Engineering, K. S. School of Engineering and Management, Bengaluru, for his encouragement.

We would like to thank our Project Guide, **Mrs Amitha S**, Associate professor and **Mrs.Deepashree N**, Assistant Professor, Department of Computer Science and Engineering, K. S. School of Engineering and Management, Bengaluru, for their constant guidance and inputs.

We would like to thank all the **Teaching** Staff and **Non-Teaching** Staff of the college for their cooperation.

Finally, we extend my heart-felt gratitude to my **family** for their encouragement and support without which we wouldn't have come so far. Moreover, we thank all my **friends** for their invaluable support and cooperation.

**AKHIL A**

**M SURABHI**

## **ABSTRACT**

The Airline Ticket Booking System streamlines airline operations, passenger management, and ticketing processes. By integrating key tables such as Airline, Airport, Flight, and Passenger, the system enables efficient scheduling, booking, and tracking of flights. With comprehensive passenger information stored in the database, including names, passport details, and contact information, the system facilitates personalized booking experiences. Additionally, the inclusion of pricing information ensures flexible fare management. Overall, the project aims to enhance airline efficiency, optimize flight operations, and deliver seamless booking experiences for passengers.

# TABLE OF CONTENTS

<b>Chapter No.</b>	<b>Contents</b>	<b>Page No.</b>
	Acknowledgement	I
	Abstract	II
	Table of Contents	III
	List of Figures	IV
	List of Tables	V
<b>Chapter 1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	OVERVIEW	1
1.2	PROBLEM STATEMENT	1
1.3	DATABASE MANAGEMENT SYSTEM	1
1.4	SQL	2
1.5	HTML / JAVASCRIPT & PHP	2
<b>Chapter 2</b>	<b>REQUIREMENTS SPECIFICATION</b>	<b>4</b>
2.1	OVERALL DESCRIPTION	4
2.2	SPECIFIC REQUIREMENTS	4
2.3	SOFTWARE REQUIREMENTS	4
2.4	HARDWARE REQUIREMENTS	4
2.5	TECHNOLOGY	5
<b>Chapter 3</b>	<b>DETAILED DESIGN</b>	<b>6</b>
3.1	SYSTEM DESIGN	6
3.2	ENTITY RELATIONSHIP DIAGRAM	7
3.3	RELATIONAL SCHEMA	9
3.4	DESCRIPTION OF TABLES	10
<b>Chapter 4</b>	<b>IMPLEMENTATION</b>	<b>12</b>
4.1	MODULE AND THEIR ROLES	12
4.2	RESULT	27
<b>Chapter 5</b>	<b>TESTING</b>	<b>28</b>
5.1	SOFTWARE TESTING	28
5.2	MODULE TESTING AND INTEGRATION	28
5.3	LIMITATIONS	29
<b>Chapter 6</b>	<b>SNAP SHOTS</b>	<b>30</b>

6.1	LOGIN PAGE FOR ADMINS	30
6.2	CUSTOMER LOGIN PAGE	30
6.3	HOME PAGE	31
6.4	LIST OF FLIGHTS	31
6.5	SEARCH FLIGHTS	32
6.6	PASSENGER DETAILS	32
6.7	MODIFY FLIGHT	33
6.8	DELETE FLIGHT	33
6.9	ADD FLIGHTS	34
6.10	DELETE TICKETS	34
6.11	MODIFY TICKETS	35
<b>Chapter 7</b>	<b>CONCLUSION</b>	<b>36</b>
<b>Chapter 8</b>	<b>FUTURE ENHANCEMENTS</b>	<b>37</b>
	<b>REFERENCES</b>	<b>38</b>

## LIST OF FIGURES

<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
<b>3.1</b>	JSP Architecture	6
<b>3.2</b>	ER diagram of Airlines Database Management System	8
<b>3.3</b>	Schema diagram	9
<b>6.1</b>	Login Page	34
<b>6.2</b>	Customer Login Page	34
<b>6.3</b>	Home Page	35
<b>6.4</b>	List of Flights	35
<b>6.5</b>	Search Flights	36
<b>6.6</b>	Passenger Details	36
<b>6.7</b>	Modify Flights	37
<b>6.8</b>	Delete Flights	37
<b>6.9</b>	Add Flights	38
<b>6.10</b>	Delete Tickets	38
<b>6.11</b>	Modify Tickets	39

## Chapter 1

# INTRODUCTION

### 1.1 OVERVIEW

An Airline Database Management System acts as the backbone of airline operations, storing and managing crucial data like flight schedules, passenger information, and booking details. It empowers airlines to streamline tasks through functionalities like flight search, booking management, and passenger check-in. This centralized data hub not only enhances customer service but also fuels data-driven decision-making, optimizing everything from pricing strategies to inventory allocation, ultimately contributing to the smooth functioning and success of the airline.

### 1.2 PROBLEM STATEMENT

This project aims to develop a robust solution that centralizes flight details, passenger data, and ticketing processes to enhance efficiency and improve customer satisfaction in the airline industry.

### 1.3 DATABASE MANAGEMENT SYSTEM

A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data. The DBMS essentially serves as an interface between the database and end users application programs, ensuring that data is consistently organized and remains easily accessible.

The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified, and the database schema, which defines the database's logical structure. These three foundational elements help to provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management, performance monitoring and back up and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

## 1.4 SQL

SQL is a standard language for storing, manipulating and retrieving data in databases.

Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language, data manipulation language ,and data control language.The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control.

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO)in 1987.Since then, the standard has been revised to include a larger set of features. Despite the existence of such standards, most SQL code is not completely portable among different database systems without adjustments.

## 1.5 HTML /Javascript & PHP

HTML is a markup language used for structuring and presenting content on the web and the fifth and current major version of the HTML standard.

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs)for complex web applications.JavaScript often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm.

Alongside HTML and CSS, JavaScript is one of the three core technologies of the World Wide Web. JavaScript enables interactive web pages and thus is an essential part of web applications. The vast majority of websites use it, and all major web browsers have a dedicated JavaScript engine to execute it.

## PHP

PHP can actually do anything related to server-side scripting or more popularly known as the backend of a website. For example, PHP can receive data from forms, generate dynamic page content, can work with databases, create sessions, send and receive cookies, send emails etc.

- 1.Simple: It is very simple and easy to use, compared to other scripting languages it is very simple and easy, this is widely used all over the world.
2. Interpreted: It is an interpreted language, i.e. there is no need for compilation.
3. Faster: It is faster than other scripting languages e.g. asp and jsp.
4. Open Source: Open source means you no need to pay to use php, you can download and use.
5. Platform Independent: PHP code will be run on every platform, Linux, UNIX, Mac OS X, and Windows.
- 6.Case Sensitive: PHP is case sensitive scripting language at time of variable declaration. In PHP, all keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are NOT case-sensitive
- 7.Error Reporting: PHP has some predefined error reporting constants to generate a warning or error notice.



## Chapter 2

### REQUIREMENTS & SPECIFICATIONS

A computerized way of handling information about property and users details is efficient, organized and time saving, compared to a manual way of doing so. This is done through a database driven web application whose requirements are mentioned in this section.

#### 2.1 OVERALL DESCRIPTION

A reliable and scalable database driven web application with security features that is easy to use and maintain is the requisite.

#### 2.2 SPECIFIC REQUIREMENTS

The specific requirements of the Airlines Database Management System are stated as follows:

#### 2.3 SOFTWARE REQUIREMENTS

- IDE - NetBeans8.2
- Web Browser – Firefox 50 or later, Google Chrome – 60 or later
- Database support - MySQL5.7
- MySQL Server 5.7
- MySQL Shell 1.0.10
- MySQLWorkbench
- Operating system – Windows 7 / Ubuntu 16.04
- JDK 1.8
- Server deployment - Xampp

#### 2.4 HARDWARE REQUIREMENTS

- Processor – Pentium IV or above
- RAM – 2 GB or more
- Hard disk – 3 GB or more
- Monitor – VGA of 1024x768 screen resolution
- Keyboard & Mouse

## 2.5 TECHNOLOGY

- HTML is used for the front end design. It provides a means to structure text based information in a document. It allows users to produce web pages that include text, graphics and hyperlinks.
- CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML ,the language can be applied to any XML document.
- SQL is the language used to manipulate relational databases. It is tied closely with the relational model. It is issued for the purpose of data definition and data manipulation.
- PHP is a popular choice for building web applications due to its simplicity, versatility, and compatibility with various operating systems and databases.

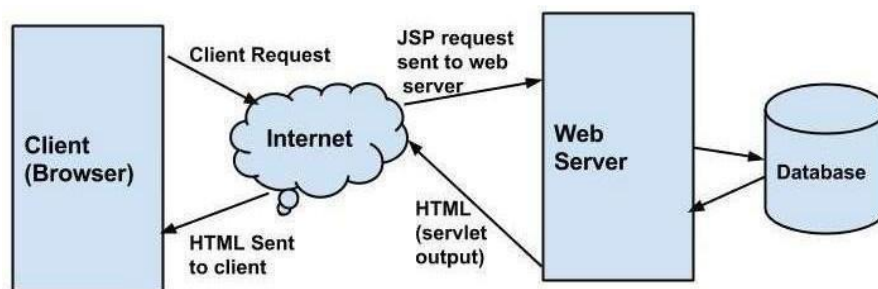
## Chapter 3

### DETAILED DESIGN

#### 3.1 SYSTEM DESIGN

The web server needs a JSP engine, i.e., a container to process JSP pages. The JSP container is responsible for intercepting requests for JSP pages. A JSP container works with the Web server to provide the runtime environment and other services a JSP needs. It knows how to understand the special elements that are part of JSPs. This server will act as a mediator between the client browser and a database.

The following diagram shows the JSP architecture.



**Fig. 3.1: JSP Architecture**

Three-tier Client / Server database architecture is commonly used architecture for web applications. Intermediate layer called Application server or Web Server stores the web connectivity software and the business logic (constraints) part of the application used to access the right amount of data from the database server. This layer acts like a medium for sending partially processed data between the database server and the client. Database architecture focuses on the design, development, implementation and maintenance of computer programs that store and organize information for businesses, agencies and institutions. A database architect develops and implements software to meet the needs of users. Several types of databases, including relational or multimedia, may be created. Additionally, database architects may use one of several languages to create databases, such as structured query language.

## 3.2 ENTITY RELATIONSHIP DIAGRAM

An entity–relationship model is usually the result of systematic analysis to define and describe what is important to processes in an area of a business.

An E-R model does not define the business processes ;it only presents a business data schema in graphical form. It is usually drawn in a graphical form as boxes(entities)that are connected by lines (relationships) which express the associations and dependencies between entities.

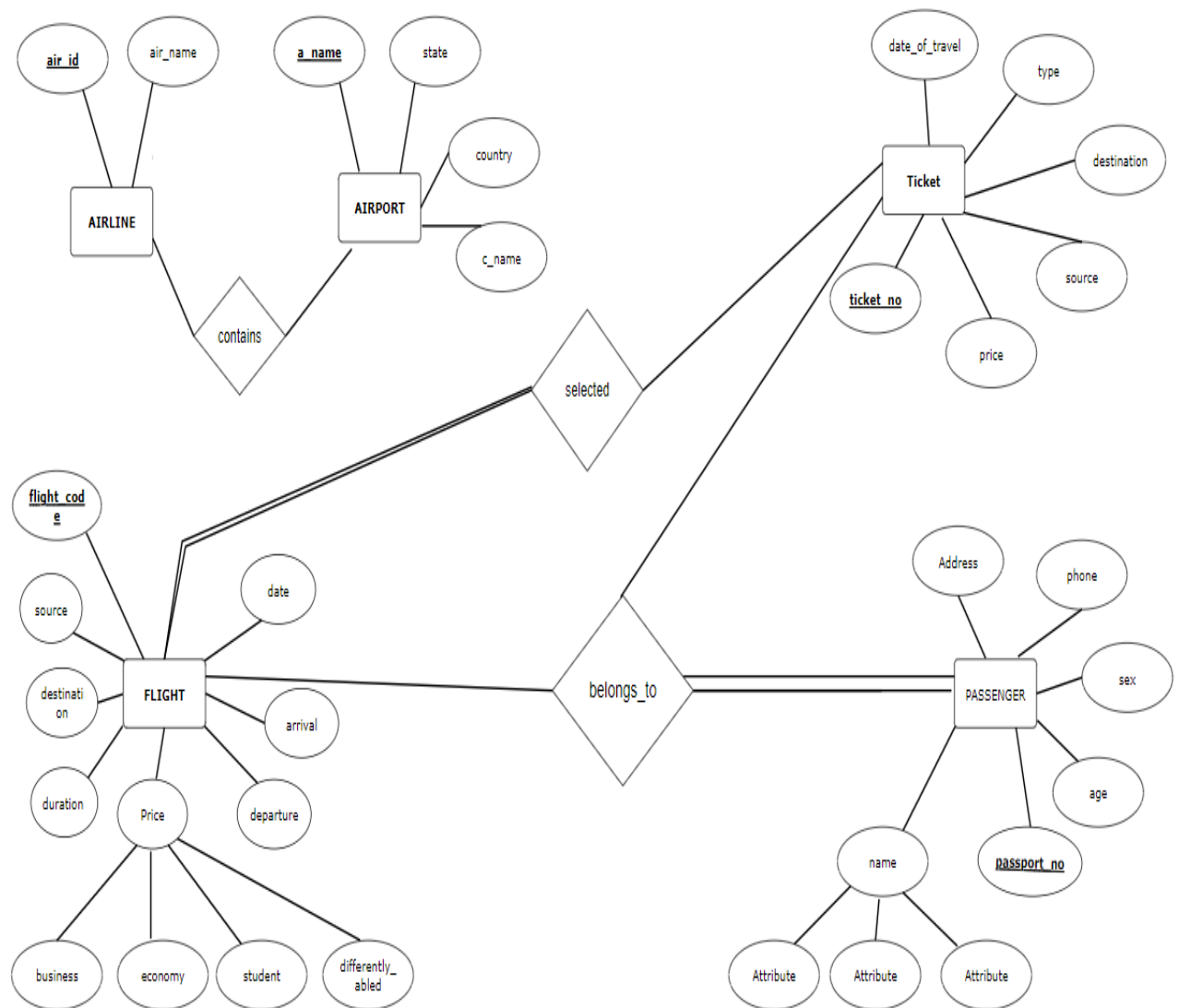
Entities may be characterized not only by relationships, but also by additional properties (attributes), which include identifiers called "primary keys". Diagrams created to represent attributes as well as entities and relationships may be called entity-attribute-relationship diagrams, rather than entity-relationship models.

An ER model is typically implemented as a database. In a simple relational database implementation, each row of a table represents one instance of an entity type, and each field in a table represents an attribute type. In a relational database a relationship between entities is implemented by storing the primary key of one entity as a pointer or "foreign key" in the table of another entity.

There is a tradition for ER/data models to be built at two or three levels of abstraction. Note that the conceptual-logical-physical hierarchy below is used in other kinds of specification, and is different from the three schema approach to software engineering. While useful for organizing data that can be represented by a relational structure, an entity-relationship diagram can't sufficiently represent semi-structured or unstructured data, and an ER Diagram is unlikely to be helpful on its own in integrating data into a pre-existing information system.

Cardinality notations define the attributes of the relationship between the entities. Cardinalities can denote that an entity is optional.

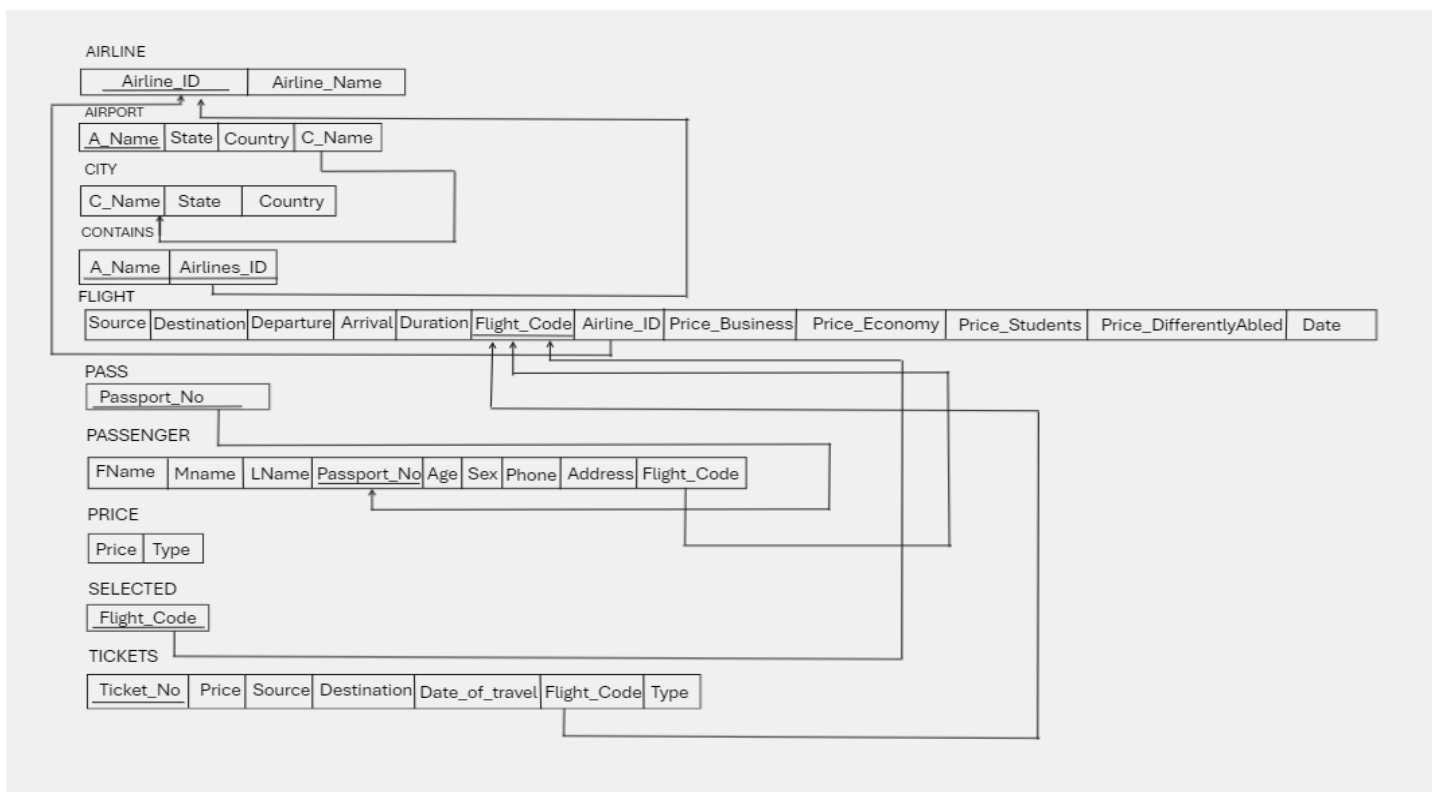
## ER DIAGRAM:



**Fig 3.2 : Entity-Relationship Diagram**

### 3.3 RELATIONAL SCHEMA

The term "schema" refers to the organization of data as a blueprint of how the database is constructed. The formal definition of a database schema is a set of formulas called integrity constraints imposed on a database. A relational schema shows references among fields in the database. When a primary key is referenced in another table in the database, it is called a foreign key. This is denoted by an arrow with the head pointing at the referenced key attribute. A schema diagram helps organize values in the database. The following diagram shows the schema diagram for the database.



**Fig. 3.3: Schema diagram**

### 3.4 DESCRIPTION OF THE TABLES

The database consists of six tables:

1.Airline: Stores information about different airlines

- AIRLINE\_ID: Unique identifier for airlines
- AIRLINE\_NAME: Name of the airline.

2.Airport: Contains details about airports, including their names, locations, and possibly associated cities.

- A\_NAME: Name of the airport.
- STATE: State where the airport is located.
- COUNTRY: Country where the airport is located.
- C\_NAME: Possibly the name of the city where the airport is located.

3.City: Stores information about cities, including their names and locations

- C\_NAME: Name of the city.
- STATE: State where the city is located.
- COUNTRY: Country where the city is located.

4.Contains: A relational table linking airports and airlines, indicating which airlines operate at which airports.

- A\_NAME:Name of the Airport.
- AIRLINE\_ID: Unique identifier for airlines.

5. Flight: Stores details about flights, including their source, destination, timings, duration, pricing, and associated airline information.

- SOURCE: Source Airport for the flight.
- DESTINATION: Destination Airport for the flight.
- DEPARTURE: Departure time of the flight.
- ARRIVAL: Arrival Time of the Flight.
- DURATION: Duration of the flight.
- FLIGHT\_CODE: Unique Identifier of the flight.
- AIRLINE\_ID: Unique identifier for airlines.
- PRICE\_BUSINESS: Price of Business Class Tickets for the flight.
- PRICE\_ECONOMY: Price of Economy Class Tickets for the flight.
- PRICE\_STUDENTS: Price of tickets for students for the flight.
- PRICE\_DIFFERENTLYABLED: Price of tickets for differently-abled passengers for the flight.
- DATE: Date of the flight.

6. PASS: To store passport information.

- PASSPORT\_NO: Passport Number.

7. PASSENGER: Stores information about passengers, including their personal details and associated flight information.

- FNAME: First Name of the passenger.
- NAME: Middle Name of the passenger.
- LNAME: Last Name of the passenger.
- PASSPORT\_NO: Passport number of the passenger.
- AGE: Age of the passenger.
- SEX: Sex of the passenger.
- PHONE: Phone number of the passenger.
- ADDRESS: Address of the passenger.
- FLIGHT\_CODE: Unique Identifier for the flight.

8. PRICE: Used to store pricing information for different types of ticket.

- PRICE: Price value.
- TYPE: Type of Price.

9. SELECTED: Used for storing selected flights during the booking process.

- FLIGHT\_CODE: Unique Identifier for the flight.

10. TICKET: Stores information about tickets, including ticket details, pricing, source, destination, travel date, passenger passport number, flight code and ticket type.

- TICKET\_NO: Unique identifier for the ticket.
- PRICE: Price of the ticket.
- SOURCE: Source airport for the ticket.
- DESTINATION: Destination airport for the ticket.
- DATE\_OF\_TRAVEL: Date of travel for the ticket.
- FLIGHT\_CODE: Unique identifier for the flight.
- TYPE: Type of the ticket.



## Chapter 4

# IMPLEMENTATION

### 4.1 MODULES AND THEIR ROLES

#### Login: Login for the admin

```
<?php
$error="";
if(isset($_POST['submit'])){
    $username=$_POST['username'];
    $password=$_POST['password'];
    if($username=="admin"){
        if($password=="admin"){
            echo
" <script>window.location='adminchoice.html'</script>";
        }
        else{
            echo "<script>alert('Invalid Password')</script>";
            echo " <script>window.location='adminlogin.php'</script>";
        }
    }
    else{
        echo "<script>alert('Invalid Username')</script>";
        echo " <script>window.location='adminlogin.php'</script>";
    }
}

?>
<!DOCTYPE html>
<html>
<head>
<title></title>
<style>
*{

    margin: 0;

    padding: 0;

    font-family: Century Gothic;

}
```

```
body{  
    background-image:linear-gradient(rgba(0,0,0,0.5),rgba(0,0,0,0.5)),  
url(plane.jpg);  
    height: 100vh;  
    background-size: cover;  
    background-position: center;  
}  
  
ul{  
    float: right;  
    list-style-type: none;  
    margin-top: 25px;  
}  
ul li{  
    display: inline-block;  
}  
ul li a{  
    text-decoration: none;  
    color: #fff;  
    padding: 5px 20px;  
    border: 1px solid #fff;  
    transition: 0.6s ease;  
}  
ul li a:hover{  
    background-color: #fff;  
    color: #000;  
}  
ul li.active a{  
    background-color: #fff;  
    color: #000;  
}  
.title{  
    position: absolute;  
    top: 30%;  
    left: 50%;  
    transform: translate(-50%,-50%);  
}  
.title h1{  
    color: #fff;  
    font-size: 70px;  
}  
table.a{
```

```
position: absolute;

top: 60%;

left: 50%;

transform: translate(-50%,-50%);

border: 1px solid #fff;

padding: 10px 30px;

color: #fff;

text-decoration: none;

transition: 0.6s ease;

font-size: 25px;

}
```

```
input[type=submit]{

border: 1px solid #fff;

padding: 10px 30px;
text-decoration: none;
transition: 0.6s ease;
}
input[type=submit]:hover{
background-color: #fff;
color: #000;
}
input[type=text],input[type=password]{
width: 100%;
padding: 12px 20px;
margin: 8px 0;
display: inline-block;
border: 1px solid #ccc;
border-radius: 4px;
box-sizing: border-box;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div class="main">
```

```
<ul>
```

```
<li class="active"><a href="#">Admin Login</a></li>
```

```
<li><a href="homepage.html">Home</a></li>
```

```
</ul>
</div>
<div class="title">
  <h1>Admin Login</h1>
</div>
<form method="post">
  <table class="a" width=40%>
    <tr>
      <td>
        User Name:
      <td>
        <input type="text" placeholder="User Name" name="username"
required>
      </td>
    </tr>
    <tr>
      <td>
        Password:
      </td>
      <td>
        <input type="password" placeholder="Password"
name="password" required>
      </td>
    </tr>
    <tr>
      <td>
        <input type="submit" name="submit"></td>
    </tr>
  </table>
</body>
</html>

<style>
  *{
    margin: 0;
    padding: 0;
    font-family: Century Gothic;
  }
  body{
    background-image:linear-gradient(rgba(0,0,0,0.5),rgba(0,0,0,0.5)),
url(plane.jpg);
    height: 100vh;
    background-size: cover;
    background-position: center;
  }
```

```
ul{
    float: right;
    list-style-type: none;
    margin-top: 25px;
}
ul li{
    display: inline-block;
}
ul li a{
    text-decoration: none;
    color: #fff;
    padding: 5px 20px;
    border: 1px solid #fff;
    transition: 0.6s ease;
}
ul li a:hover{
    background-color: #fff;
    color: #000;
}
ul li.active a{
    background-color: #fff;
    color: #000;
}
}
.title{
    position: absolute;
    top: 30%;
    left: 50%;
    transform: translate(-50%,-50%);
}
.title h1{
    color: #fff;
    font-size: 70px;
}
}
table.a{
    position: absolute;
    top: 60%;
    left: 50%;
    transform: translate(-50%,-50%);
    border: 1px solid #fff;

    padding: 10px 30px;

    color: #fff;

    text-decoration: none;

    transition: 0.6s ease;
```

font-size: 25px;

}

```
input[type=submit]{
    border: 1px solid #fff;
    padding: 10px 30px;
    text-decoration: none;
    transition: 0.6s ease;
```

}

```
input[type=submit]:hover{
    background-color: #fff;
    color: #000;
```

}

```
input[type=text],input[type=password]{
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    display: inline-block;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}
```

</style>

</head>

<body>

<div class="main">

<ul>

<li class="active"><a href="#">Admin Login</a></li>

<li><a href="homepage.html">Home</a></li>

</ul>

</div>

<div class="title">

<h1>Admin Login</h1>

</div>

<form method="post">

<table class="a" width=40%>

<tr>

<td>

User Name:

<td>

<input type="text" placeholder="User Name" name="username" required>

</td>

</tr>

<tr>

<td>

Password:

</td>

```

        <td>
            <input type="password" placeholder="Password"
name="password" required>
        </td>
    </tr>
    <tr>
        <td>
            <input type="submit" name="submit">
        </td>
    </tr>
</body></html>

```

### Search Flight:

```

<?php
require_once "dbconnection.php";
if(isset($_POST['search']))
{
    if(isset($_POST['source']) && !empty($_POST['source']))
    {
        if(isset($_POST['destination']) && !empty($_POST['destination']))
        {
            if(isset($_POST['date']) && !empty($_POST['date']))
            {
                $source=$_POST['source'];
                $destination=$_POST['destination'];
                $date=$_POST['date'];
                $query=mysqli_query($con,("select
ARRIVAL,DEPARTURE,DURATION,FLIGHT_CODE,AIRLINE_ID,PRICE_BUSINESS,PRI
CE_ECONOMY,PRICE_STUDENTS,PRICE_DIFFERENTLYABLED from flight where
SOURCE='$source' && DESTINATION='$destination' && DATE='$date'"));
                $rowscount=mysqli_num_rows($query);
                if ($rowscount==0){
                    echo "<script>alert('No Flights available')</script>";
                    echo
"<script>window.location='searchflight.html'</script>";
                }
            }
            else{
                echo "<script>alert('Please Enter the details correctly')</script>";
                echo "<script>window.location='homepage.html'</script>";
            }
        }
    }
}

```

```
    }
    else{
        echo "<script>alert('Please Enter the details correctly')</script>";
        echo "<script>window.location='homepage.html'</script>";
    }
}
else{
    echo "<script>alert('Please Enter the details correctly')</script>";
    echo "<script>window.location='homepage.html'</script>";
}
}
?>
<!DOCTYPE html>
<html>
<head>
<title></title>
<style>
    *{
        margin: 0;
        padding: 0;
        font-family: Century Gothic;
    }
    ul{
        float: right;
        list-style-type: none;
        margin-top: 25px;
    }
    ul li{
        display: inline-block;
    }
    ul li a{
        text-decoration: none;
        color: #fff;
        padding: 5px 20px;
        border: 1px solid #fff;
        transition: 0.6s ease;
    }
    ul li a:hover{
        background-color: #fff;
        color: #000;
    }
    ul li.active a{
        background-color: #fff;
        color: #000;
    }
    .title{
        position: absolute;
        top: 15%;
```



```
        left: 35%;
        /*transform: translate(-50%,-50%);*/
    }
    .title h1{
        color: #fff;
        font-size: 70px;
    }
    body{
        background-image:linear-gradient(rgba(0,0,0,0.5),rgba(0,0,0,0.5)), url(plane.jpg);
        height: 100vh;
        background-size: cover;
        background-position: center;
    }
    table.a{
        position: absolute;
        top: 27%;
        left: 20%;
        /*transform: translate(-50%,-50%);*/
        border: 1px solid #fff;
        padding: 10px 30px;
        color: #fff;
text-decoration: none;
        transition: 0.6s ease;
        font-size: 20px;
    }
    button[type="submit"]{
        border: 1px solid #fff;
        padding: 10px 30px;
        text-decoration: none;
        transition: 0.6s ease;
    }
    button[type="submit"]:hover{
        background-color: #fff;
        color: #000;
    }
</style>
</head>
<body>
<div class="main">
    <ul>
        <li class="active"><a href="#">Available Flights</a></li>
        <li><a href="homepage.html">Home</a></li>
    </ul>
</div>
<div class="title">
    <h1>Available Flights</h1>
</div>
<table class="a">
```

---

```
<tr>
    <th>DEPARTURE&emsp;</th>
    <th>ARRIVAL&emsp;</th>
    <th>DURATION&emsp;</th>
    <th>FLIGHT_CODE&emsp;</th>
    <th>AIRLINE_ID&emsp;</th>
    <th>PRICE&emsp;</th>
    <th>TYPE&emsp;</th>
    <th></th>
</tr>

<form action="postflightcode.php" method="post">
<?php
    for($i=1;$i<=$rowscount;$i++)
    {
        $rows=mysqli_fetch_array($query);
?>
        <tr>
            <td><?php echo $rows['DEPARTURE'] ?></td>
            <td><?php echo $rows['ARRIVAL'] ?></td>
            <td><?php echo $rows['DURATION'] ?></td>
            <td><?php echo $rows['FLIGHT_CODE']?></td>
            <td><?php echo $rows['AIRLINE_ID']?></td>
            <td><?php echo $rows['PRICE_BUSINESS']?></td>
            <td><?php echo "BUSINESS";?></td>
            <td>&nbsp;<button type="submit"><a
href="postflightcodebusiness.php?id=<?php echo $rows['FLIGHT_CODE']
?>">Select</a></button></td>
        </tr>
        <tr>
            <td><?php echo $rows['DEPARTURE'] ?></td>
            <td><?php echo $rows['ARRIVAL'] ?></td>
            <td><?php echo $rows['DURATION'] ?></td>
            <td><?php echo $rows['FLIGHT_CODE']?></td>
            <td><?php echo $rows['AIRLINE_ID']?></td>
            <td><?php echo $rows['PRICE_ECONOMY']?></td>
            <td><?php echo "ECONOMY";?></td>
            <td>&nbsp;<button type="submit"><a
href="postflightcodeeconomy.php?id=<?php echo $rows['FLIGHT_CODE']
?>">Select</a></button></td>
        </tr>
        <tr>
            <td><?php echo $rows['DEPARTURE'] ?></td>
            <td><?php echo $rows['ARRIVAL'] ?></td>
            <td><?php echo $rows['DURATION'] ?></td>
            <td><?php echo $rows['FLIGHT_CODE']?></td>
            <td><?php echo $rows['AIRLINE_ID']?></td>
            <td><?php echo $rows['PRICE_STUDENTS']?></td>
```

```

        <td><?php echo "STUDENTS";?></td>
        <td>&nbsp;<button type="submit"><a
href="postflightcodestudents.php?id=<?php echo $rows['FLIGHT_CODE']
?>">Select</a></button></td>
    </tr>
    <tr>
        <td><?php echo $rows['DEPARTURE'] ?></td>
        <td><?php echo $rows['ARRIVAL'] ?></td>
        <td><?php echo $rows['DURATION'] ?></td>
        <td><?php echo $rows['FLIGHT_CODE']?></td>
        <td><?php echo $rows['AIRLINE_ID']?></td>
        <td><?php echo
$rows['PRICE_DIFFERENTLYABLED']?></td>
        <td><?php echo "DIFFERENTLY ABLED";?></td>
        <td>&nbsp;<button type="submit"><a
href="postflightcodediff.php?id=<?php echo $rows['FLIGHT_CODE']
?>">Select</a></button></td>
    </tr>
    <?php
    }
    ?>
</form>
</table>
</body>
</html>

```

```

View Flight:<?php
require_once "dbconnection.php";
?>
<!DOCTYPE html>
<html>
<head>
<title></title>
<style>
    *{
        margin: 0;
        padding: 0;
        font-family: Century Gothic;
    }
    ul{
        float: right;
        list-style-type: none;
        margin-top: 25px;
    }
    ul li{
        display: inline-block;
    }

```

```
ul li a{
  text-decoration: none;
  color: #fff;
  padding: 5px 20px;
  border: 1px solid #fff;
  transition: 0.6s ease;
}
ul li a:hover{
  background-color: #fff;
  color: #000;
}
ul li.active a{
  background-color: #fff;
  color: #000;
}
.title{
  position: absolute;
  top: 15%;
  left: 40%;
  /*transform: translate(-50%,-50%);*/
}
.title h1{
  color: #fff;
  font-size: 70px;
}
body{
  background-image:linear-gradient(rgba(0,0,0,0.5),rgba(0,0,0,0.5)), url(plane.jpg);
  height: 100vh;
  background-size: cover;
  background-position: center;
}
table.a{
  position: absolute;
  top: 27%;
  left: 0%;
  /*transform: translate(-50%,-50%);*/
  border: 1px solid #fff;
  padding: 10px 30px;
  color: #fff;
  text-decoration: none;
  transition: 0.6s ease;
  font-size: 18px;
}
button[type="submit"]{
  border: 1px solid #fff;
  padding: 10px 30px;
  text-decoration: none;
  transition: 0.6s ease;
```

```
    }
    button[type="submit"]:hover{
        background-color: #fff;
        color: #000;
    }
</style>
</head>
<body>
<div class="main">
    <ul>
        <li class="active"><a href="#">All Flights</a></li>
        <li><a href="adminchoice.html">Admin</a></li>
        <li><a href="homepage.html">Home</a></li>
    </ul>
</div>
<div class="title">
    <h1>All Flights</h1>
</div>
<table class="a">
    <tr>
        <th>SOURCE&emsp;</th>
        <th>DESTINATION&emsp;</th>
        <th>DEPARTURE&emsp;</th>
        <th>ARRIVAL&emsp;</th>
        <th>DURATION&emsp;</th>
        <th>FLIGHT_CODE&emsp;</th>
        <th>AIRLINE_ID&emsp;</th>
        <th>PRICE(BUSINESS)&emsp;</th>
        <th>PRICE(ECONOMY)&emsp;</th>
        <th>PRICE(STUDENT)&emsp;</th>
        <th>PRICE(DIFF)&emsp;</th>
        <th>DATE&emsp;</th>
        <th></th>
    </tr>
<tr></tr>
<form>
<?php
    $query=mysqli_query($con,"select * from flight");
    $rowscount=mysqli_num_rows($query);
    for($i=1;$i<=$rowscount;$i++)
    {
        $rows=mysqli_fetch_array($query);
?>
        <tr>
            <td><?php echo $rows['SOURCE'] ?></td>
            <td><?php echo $rows['DESTINATION'] ?></td>
            <td><?php echo $rows['DEPARTURE'] ?></td>
            <td><?php echo $rows['ARRIVAL'] ?></td>
```

```

        <td><?php echo $rows['DURATION'] ?></td>
        <td><?php echo $rows['FLIGHT_CODE']?></td>
        <td><?php echo $rows['AIRLINE_ID']?></td>
        <td><?php echo $rows['PRICE_BUSINESS']?></td>
        <td><?php echo $rows['PRICE_ECONOMY']?></td>
        <td><?php echo $rows['PRICE_STUDENTS']?></td>
        <td><?php echo
$rows['PRICE_DIFFERENTLYABLED']?></td>
        <td><?php echo $rows['DATE'] ?></td>
    </tr>
<?php
}
?>
</form>
</table>
</body>
</html>

```

### Passenger Details:

```

<?php
require_once "dbconnection.php";
if(isset($_POST['submit']))
{
    $condition=0;
    $len=0;
    $firstname=$_POST['firstname'];
    $middlename=$_POST['middlename'];
    $lastname=$_POST['lastname'];
    $query=mysqli_query($con,("select * from selected"));
    $rows=mysqli_fetch_array($query);
    $flight=$rows['FLIGHT_CODE'];
    $age=$_POST['age'];
    $sex=$_POST['Sex'];
    $phonenum=$_POST['phonenum'];
    $address=$_POST['address'];
    $passportnumber=$_POST['passportnumber'];
    $len= strlen($passportnumber);
    $sql1=mysqli_query($con,"select PRICE from price");
    $row=mysqli_fetch_array($sql1);
    $price=$row['PRICE'];
    $sql="select * from passenger where PASSPORT_NO='".$passportnumber"'";
    $res=mysqli_query($con,$sql);
    if($len==8){
        if(strlen($phonenum)==10){
            if(mysqli_num_rows($res)>0){
                while($row=mysqli_fetch_assoc($res))
                {
                    if($passportnumber==$row['PASSPORT_NO'])

```

```
{
    $query=mysqli_query($con,("select * from selected"));
    $rows=mysqli_fetch_array($query);
    $flight=$rows['FLIGHT_CODE'];
    $sql1=mysqli_query($con,"DELETE FROM selected WHERE
FLIGHT_CODE='$flight'");
    $sql2=mysqli_query($con,"DELETE FROM price WHERE
PRICE='$price'");
    $condition=1;
    echo "<script>alert('Enter Unique Passport Number')</script>";
    echo "<script>window.location='searchflight.html'</script>";
}
else{
    $condition=0;
}
}}
if($condition==0){
    $sql="insert into
passenger(FNAME,MNAME,LNAME,PASSPORT_NO,AGE,SEX,PHONE,ADDRESS,FLIGH
T_CODE)
values('$firstname','$middlename','$lastname','$passportnumber','$age','$sex','$phonenumner','$ad
dress','$flight')";
    mysqli_query($con,$sql);
    $passportnumber=$_POST['passportnumber'];
    $sql="insert into pass(PASSPORT_NO) values('$passportnumber')";
    mysqli_query($con,$sql);
    $query=mysqli_query($con,("select * from selected"));
    $rows=mysqli_fetch_array($query);
    $flight=$rows['FLIGHT_CODE'];
    $query=mysqli_query($con,("select SOURCE,DESTINATION,DATE from flight
where FLIGHT_CODE='$flight'"));
    $rows1=mysqli_fetch_array($query);
    $source=$rows1['SOURCE'];
    $destination=$rows1['DESTINATION'];
    $date=$rows1['DATE'];
    $query=mysqli_query($con,("select PRICE,TYPE from price"));
    $rows2=mysqli_fetch_array($query);
    $price=$rows2['PRICE'];
    $type=$rows2['TYPE'];
    $sql="insert into
ticket(PRICE,SOURCE,DESTINATION,DATE_OF_TRAVEL,PASSPORT_NO,FLIGHT_COD
E,TYPE) values('$price','$source','$destination','$date','$passportnumber','$flight','$type')";
    if(mysqli_query($con,$sql))
    {
        echo "<script>alert('Ticket Booked Successfully')</script>";
        echo "<script>window.location='reviewticket.php'</script>";
    }
    else
```

```
        {
            echo "<script>alert('Ticket Booking Failed')</script>";
        }
    }
}
else{
    $query=mysqli_query($con,("select * from selected"));
    $rows=mysqli_fetch_array($query);
    $flight=$rows['FLIGHT_CODE'];
    $sql1=mysqli_query($con,"DELETE FROM selected WHERE
FLIGHT_CODE='$flight'");
    $sql2=mysqli_query($con,"DELETE FROM price WHERE PRICE='$price'");
    echo "<script>alert('Phone Number should be of 10 digits !')</script>";
    echo "<script>window.location='searchflight.html'</script>";
}
}}
else{
    $query=mysqli_query($con,("select * from selected"));
    $rows=mysqli_fetch_array($query);
    $flight=$rows['FLIGHT_CODE'];
    $sql1=mysqli_query($con,"DELETE FROM selected WHERE
FLIGHT_CODE='$flight'");
    $sql2=mysqli_query($con,"DELETE FROM price WHERE PRICE='$price'");
    echo "<script>alert('Passport Number should be of 8 digits !')</script>";
    echo "<script>window.location='searchflight.html'</script>";
}
}
?>
```

## 4.2 RESULT

The resulting system is able to:

- Display the available flights.
- Modify the flights.
- Delete the flights.
- Provide customer login.
- Allow the customer to book a ticket.
- Allow the customer to modify the ticket.
- Allow the customer to delete the ticket.



## **Chapter 5**

# **TESTING**

### **5.1 SOFTWARE TESTING**

Testing is the process used to help identify correctness, completeness, security and quality of developed software. This includes executing a program with the intent of finding errors. It is important to distinguish between faults and failures. Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors. It can be conducted as soon as executable software (even if partially complete) exists. Most testing occurs after system requirements have been defined and then implemented in testable programs.

### **5.2 MODULE TESTING AND INTEGRATION**

Module testing is a process of testing the individual subprograms, subroutines, classes, or procedures in a program. Instead of testing the whole software program at once, module testing recommends testing the smaller building blocks of the program. It is largely white box oriented. The objective of doing Module testing is not to demonstrate proper functioning of the module but to demonstrate the presence of an error in the module. Module testing allows implementation of parallelism into the testing process by giving the opportunity to test multiple modules simultaneously.

The final integrated system too has been tested for various test cases such as duplicate entries and type mismatch.

### 5.3 LIMITATIONS

- Does not have the ability to compare ticket price to get the cheapest.
- Only restricted to certain sources and destinations.
- Only restricted to certain domestic & international airlines.
- Does not have the ability to show the final ticket once the customer books the ticket.
- Does not have the ability to allow customers to book seat of their choice.

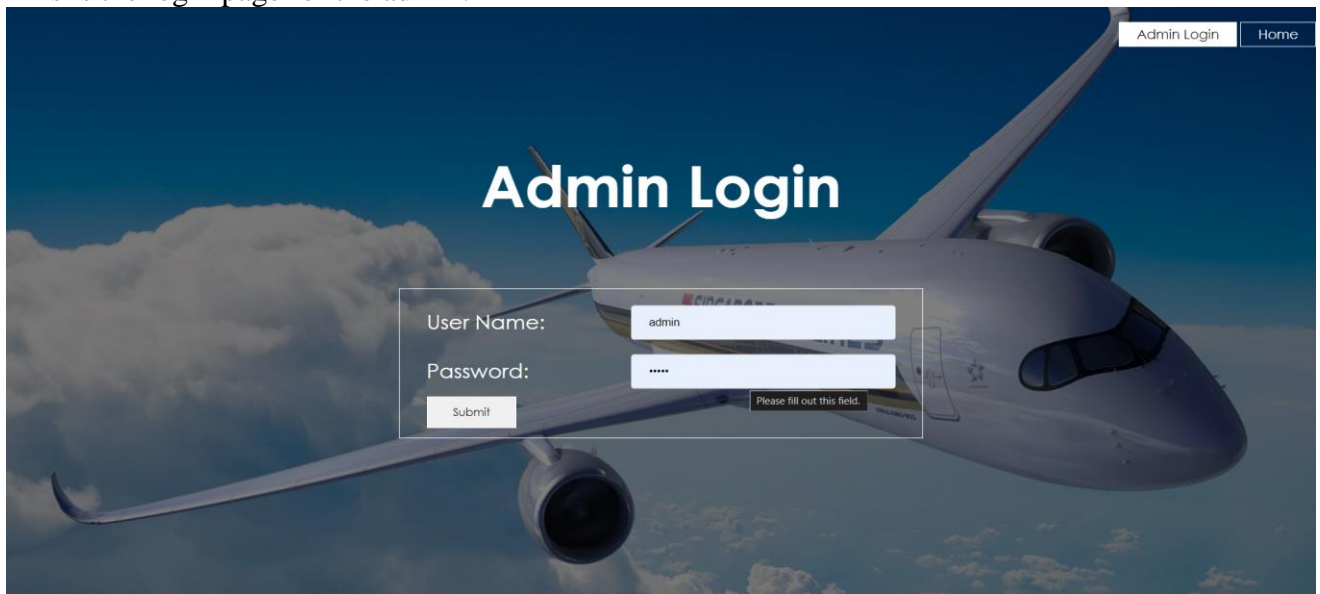
## Chapter 6

### SNAPSHOTS

This chapter consists of working screenshots of the project.

#### 6.1 LOGIN PAGE FOR ADMINS:

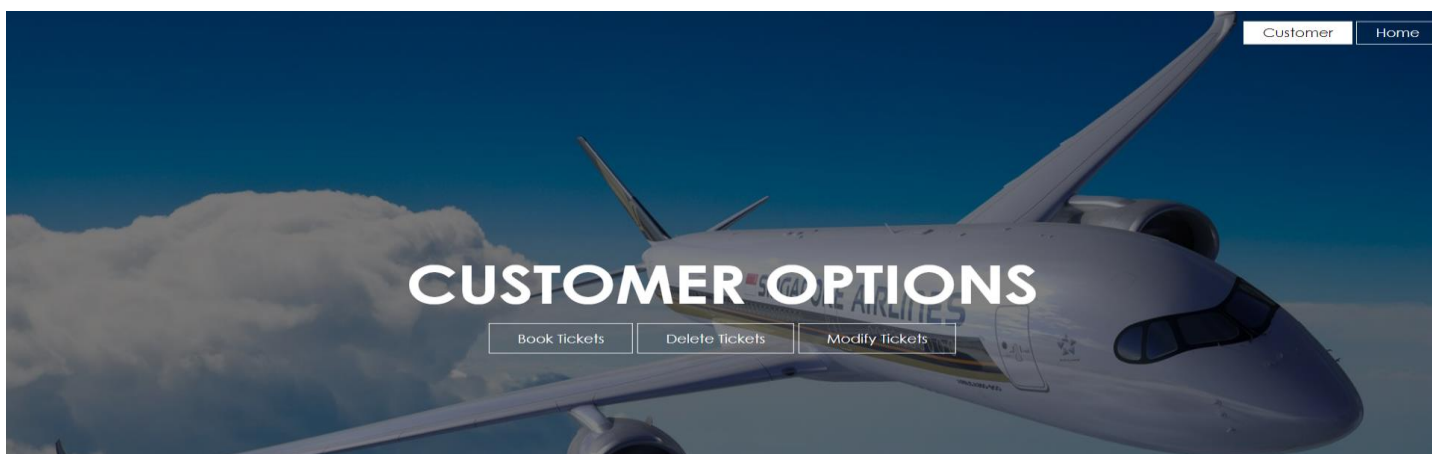
This is the login page for the admin.



**Fig 6.1:Login page**

#### 6.2 CUSTOMER LOGIN PAGE:

This is the customer login page:



**Fig 6.2: Customer Login Page**

### 6.3 HOME PAGE :

First home page shown.

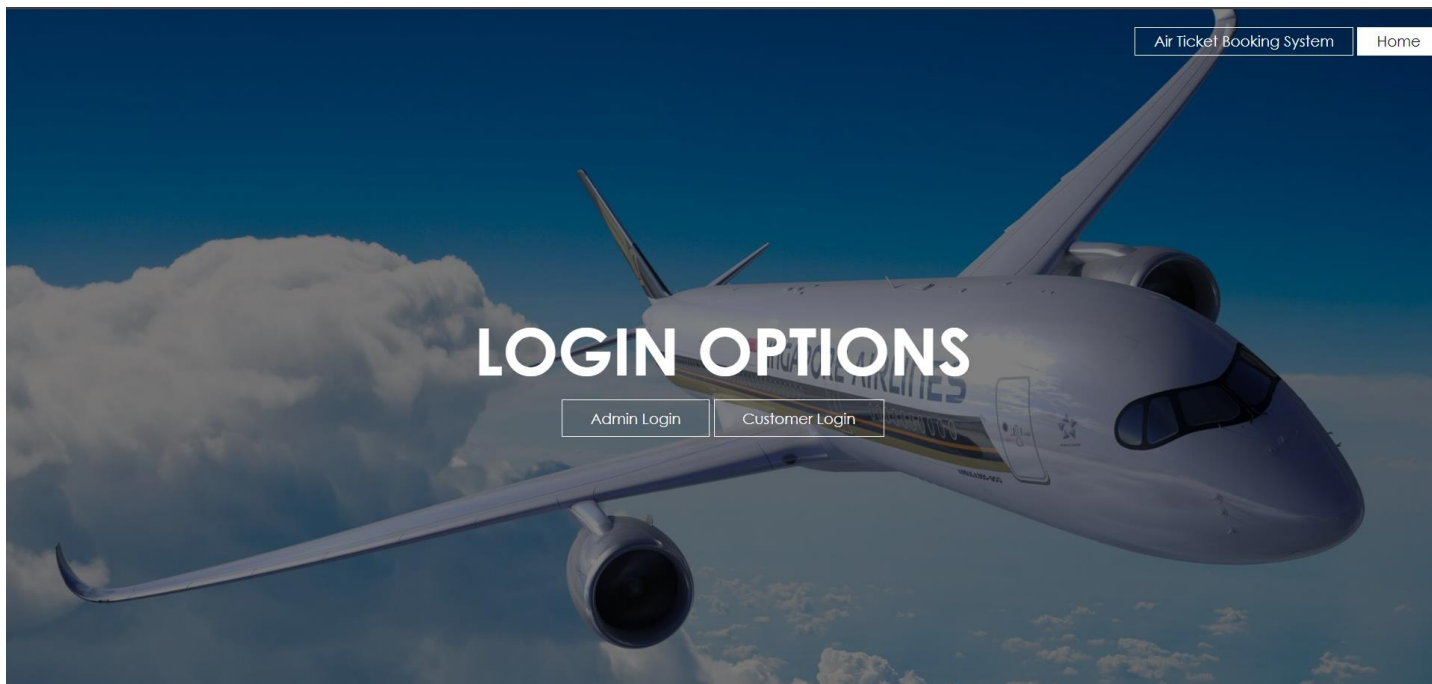


Fig 6.3: Home Page

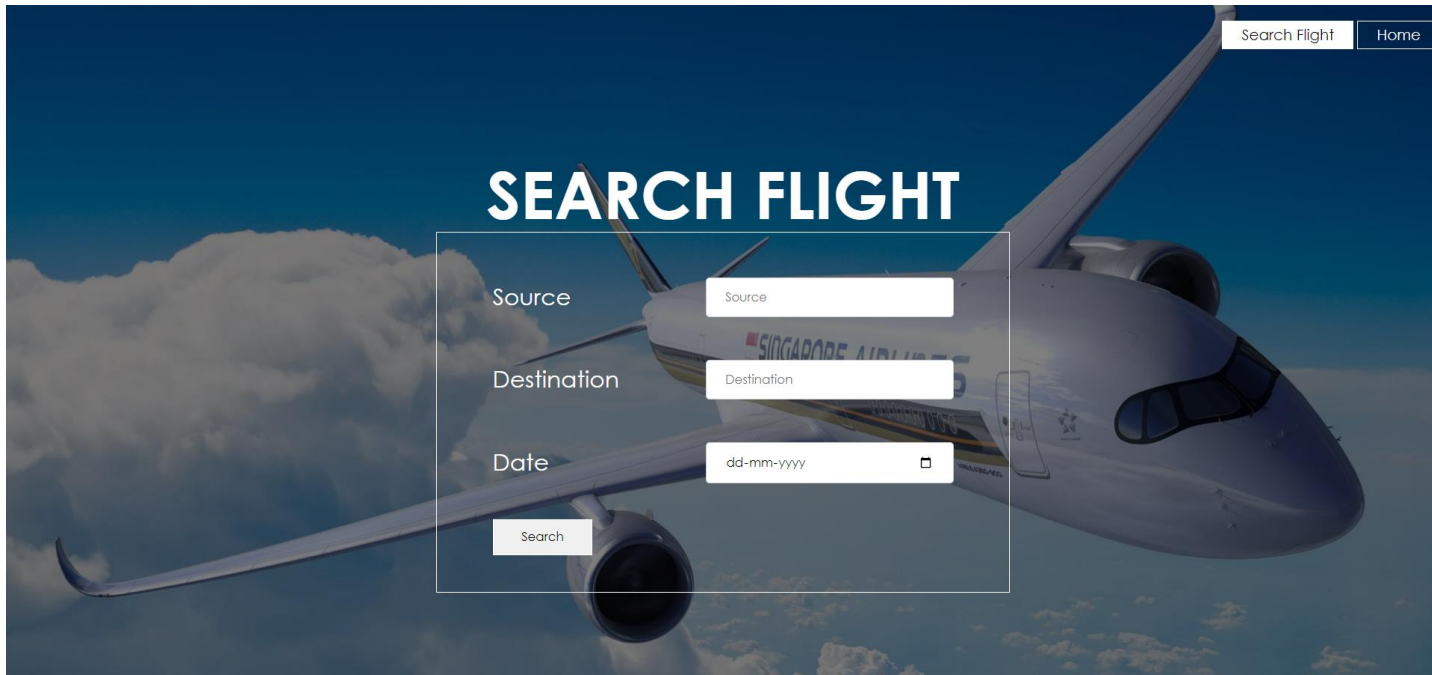
### 6.4 LIST OF FLIGHTS: List of flights available:

SOURCE	DESTINATION	DEPARTURE	ARRIVAL	DURATION	FLIGHT_CODE	AIRLINE_ID	PRICE(BUSINESS)	PRICE(ECONOMY)	PRICE(STUDENT)	PRICE(DIFF)	DATE
Bangalore	Mumbai	5:40	7:40	2 Hou	aa201	a111	22000	3722	2500	1800	2024-01-16
Chennai	Bangalore	12:30	1:30	1 Hou	aa202	a111	18601	1750	1300	800	2024-02-14
Mumbai	Hyderabad	4:00	5:25	1 Hou	aa203	a111	17849	3625	2800	2200	2024-03-03
Bangalore	Chennai	2:00	3:00	1 Hou	ai301	a112	18601	2154	1800	1100	2024-04-26

Fig 6.4: List of Flights

## 6.5 SEARCH FLIGHT

Allows customers to search the flights for their choice of source & destination along with the date.

The screenshot shows a web interface for searching flights. The background is a high-quality image of a Singapore Airlines aircraft in flight against a blue sky with white clouds. In the top right corner, there are two navigation links: "Search Flight" and "Home". The main heading "SEARCH FLIGHT" is centered in large, white, bold, sans-serif capital letters. Below the heading is a white rectangular form with a thin grey border. Inside the form, there are three input fields: "Source" with the placeholder text "Source", "Destination" with the placeholder text "Destination", and "Date" with the placeholder text "dd-mm-yyyy" and a small calendar icon to its right. At the bottom left of the form is a white "Search" button.

Search Flight Home

# SEARCH FLIGHT

Source

Destination

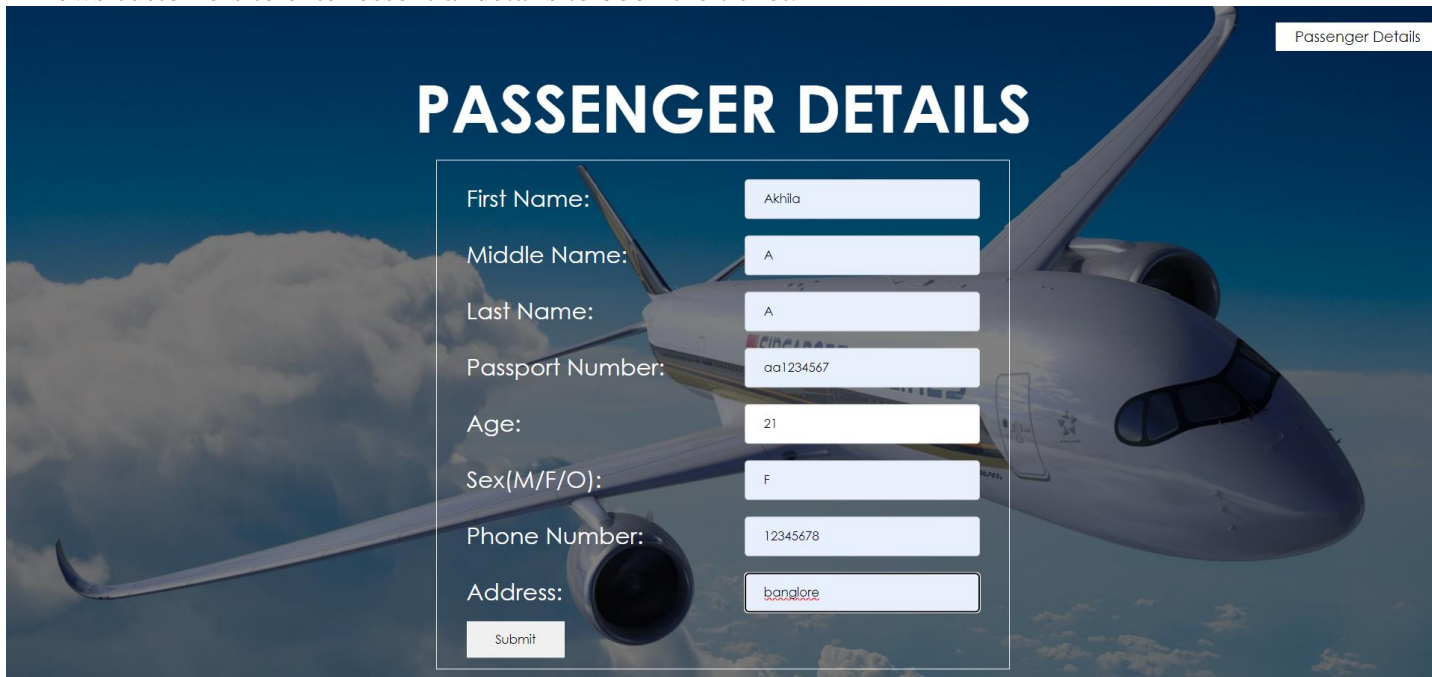
Date

Search

**Fig 6.5: Search Flight**

## 6.6 PASSENGER DETAILS

Allows customers to enter essential details to book the ticket.

The screenshot shows a web interface for entering passenger details. The background is the same Singapore Airlines aircraft image as in the previous figure. In the top right corner, there is a navigation link: "Passenger Details". The main heading "PASSENGER DETAILS" is centered in large, white, bold, sans-serif capital letters. Below the heading is a white rectangular form with a thin grey border. Inside the form, there are seven input fields: "First Name:" with the value "Akhila", "Middle Name:" with the value "A", "Last Name:" with the value "A", "Passport Number:" with the value "aa1234567", "Age:" with the value "21", "Sex(M/F/O):" with the value "F", and "Phone Number:" with the value "12345678". Below these is an "Address:" field with the value "bangalore" in red text. At the bottom left of the form is a white "Submit" button.

Passenger Details

# PASSENGER DETAILS

First Name:

Middle Name:

Last Name:

Passport Number:

Age:

Sex(M/F/O):

Phone Number:

Address:

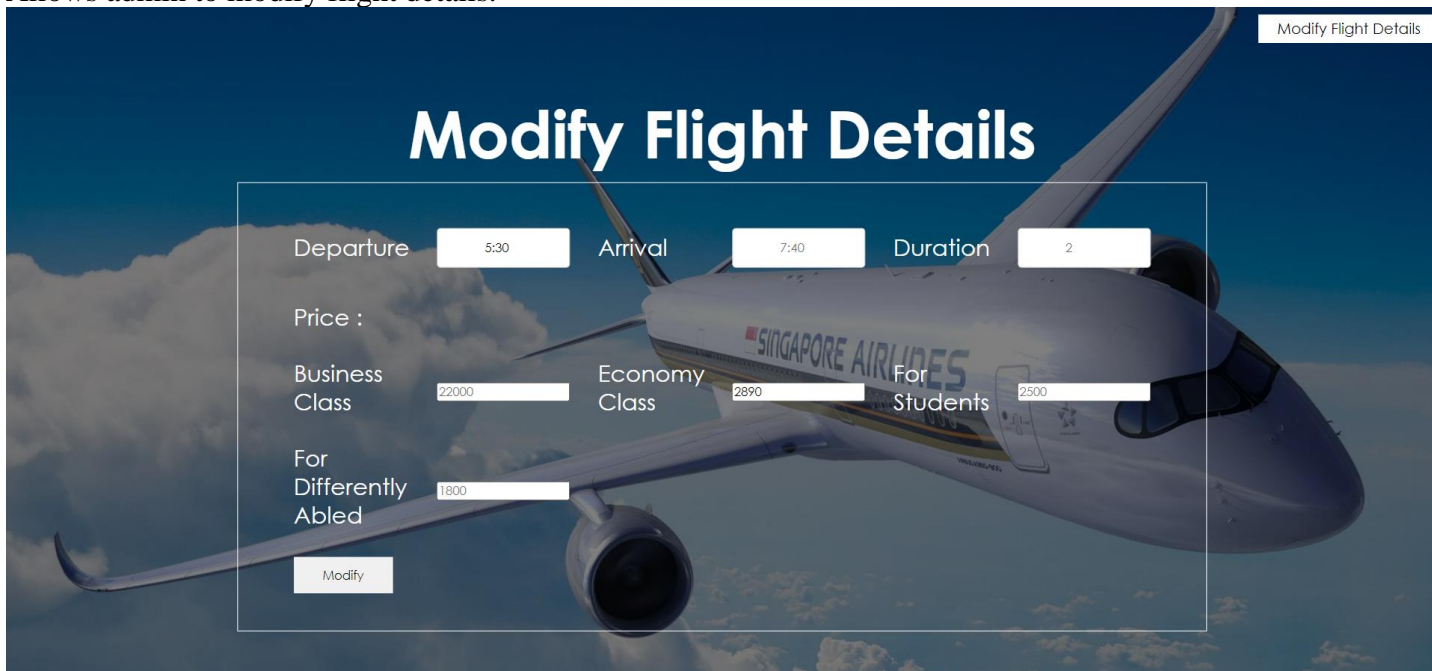
Submit

**Fig 6.6: Passenger Details**



## 6.7 MODIFY FLIGHTS

Allows admin to modify flight details.



Modify Flight Details

Departure  Arrival  Duration

Price :

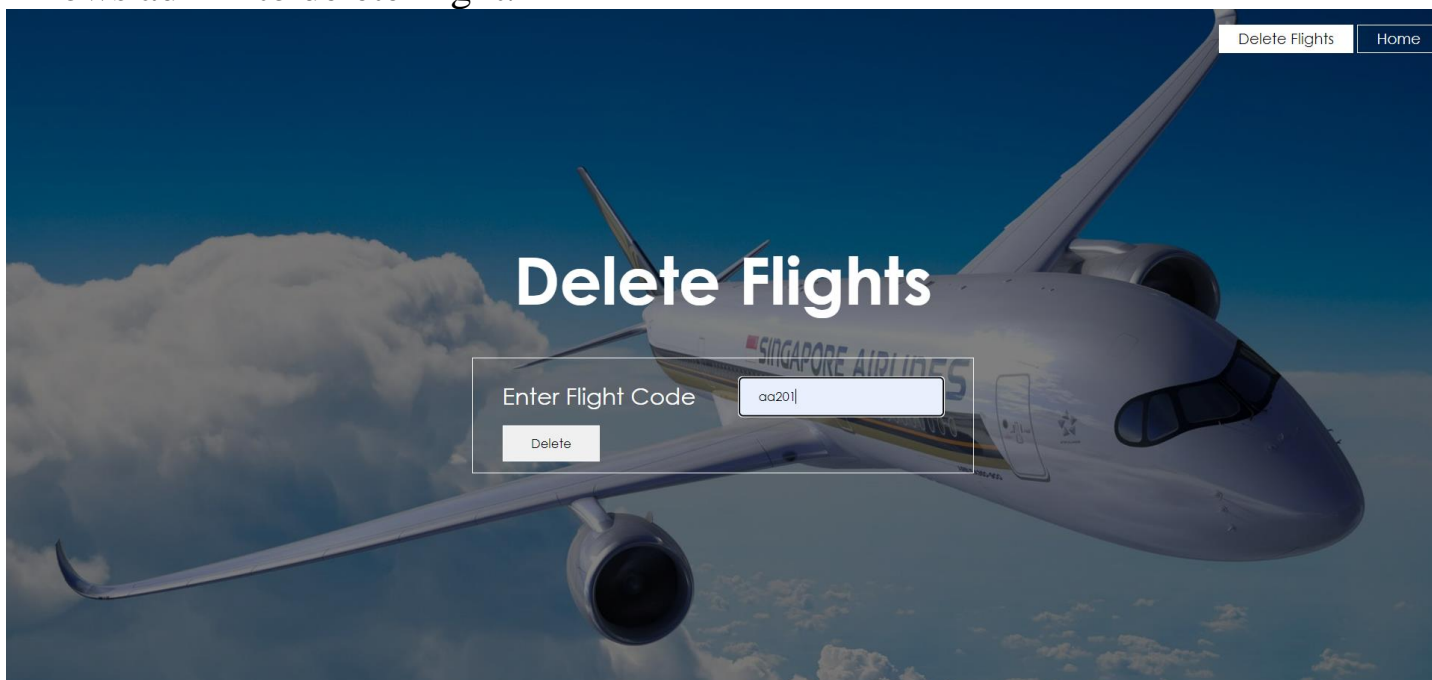
Business Class  Economy Class  For Students

For Differently Abled

**Fig 6.7: Modify Flights**

## 6.8 DELETE FLIGHT

Allows admin to delete flight.



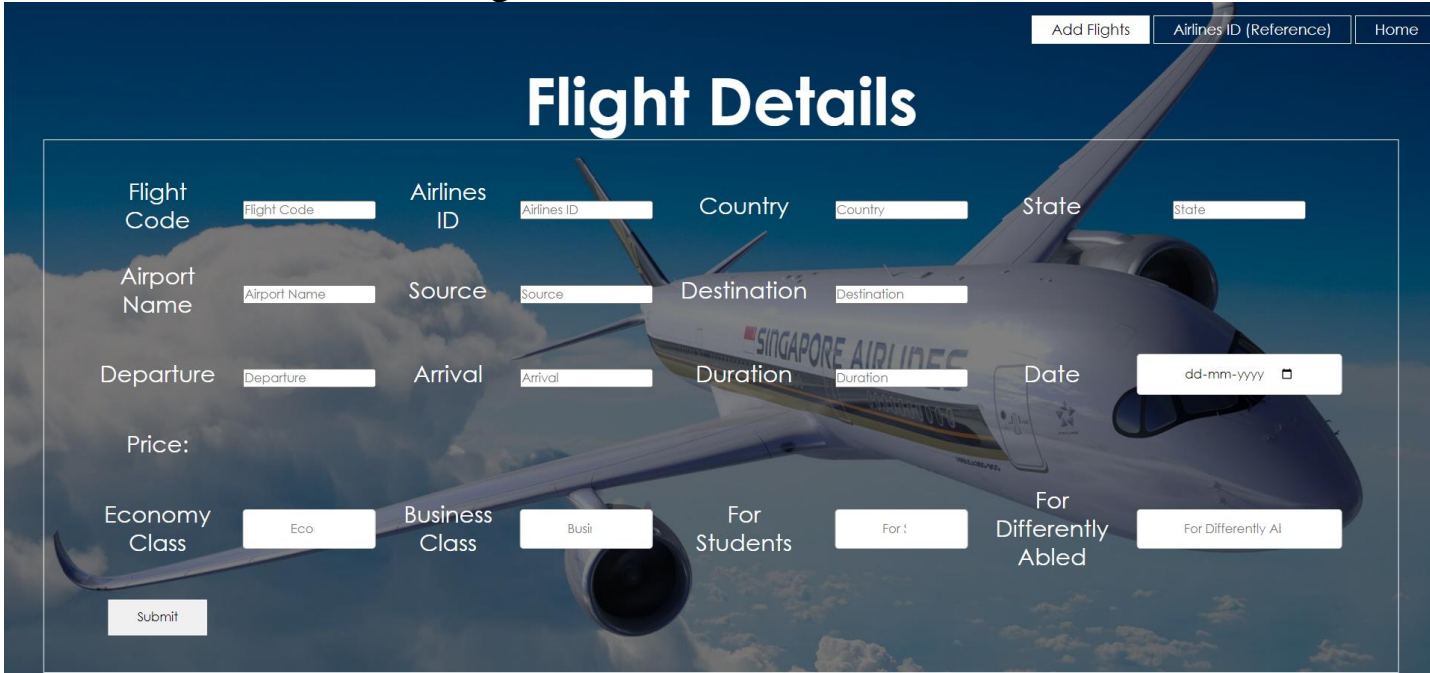
Delete Flights Home

Enter Flight Code

**Fig 6.8:Delete Flight**

## 6.9 ADD FLIGHT

Allows admin to add new flights.



**Flight Details**

Flight Code  Airlines ID  Country  State

Airport Name  Source  Destination

Departure  Arrival  Duration  Date

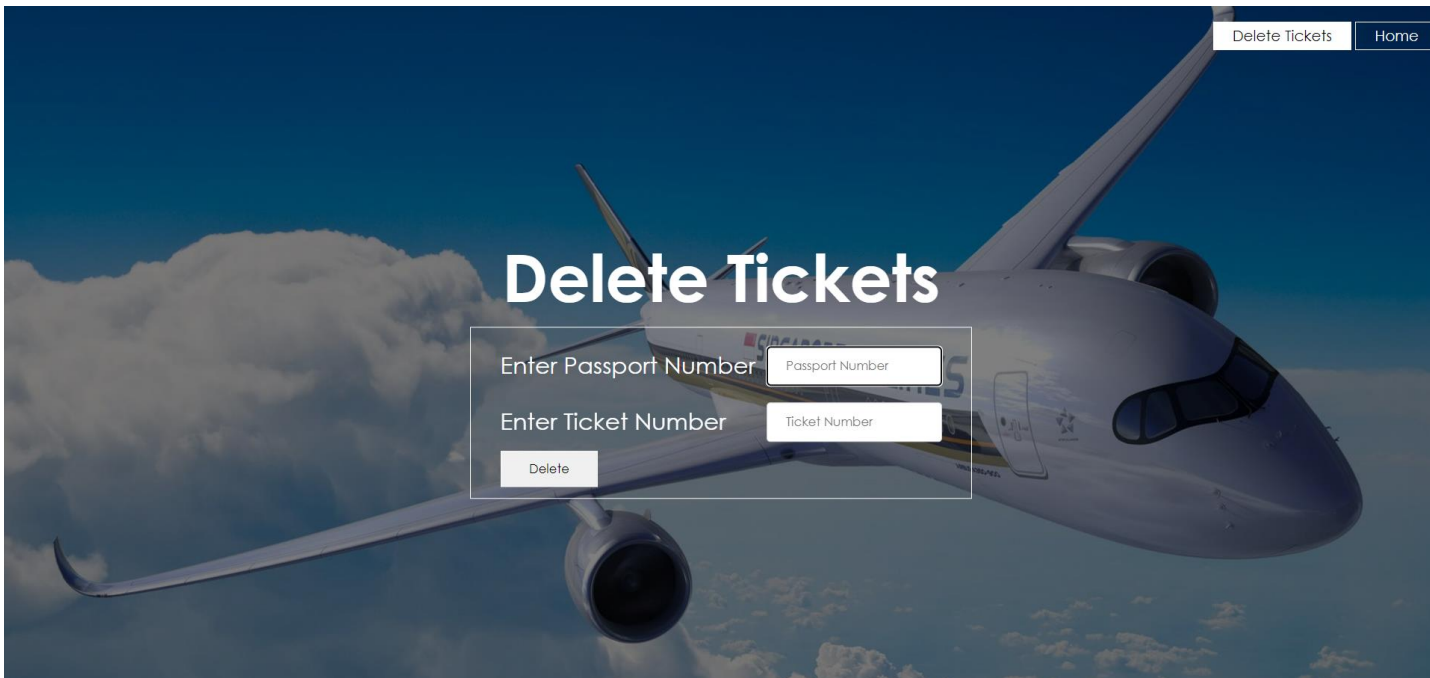
Price:

Economy Class  Business Class  For Students  For Differently Abled

**Fig 6.9: Add flights.**

## 6.10 DELETE TICKETS

Allows customers to delete tickets.



**Delete Tickets**

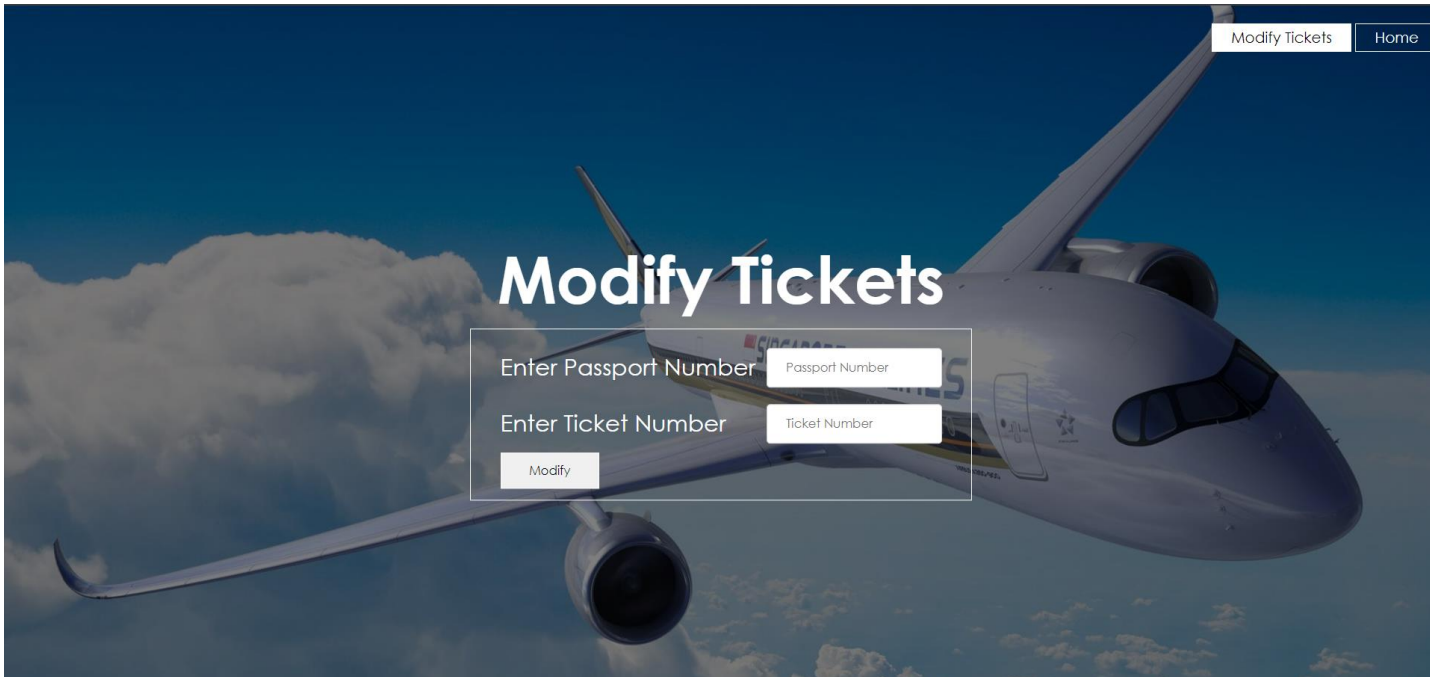
Enter Passport Number

Enter Ticket Number

**Fig 6.10: Delete Ticket**

## 6.11 MODIFY TICKET

Allows customers to modify tickets.



Modify Tickets

Home

Modify Tickets

Enter Passport Number

Enter Ticket Number

**Fig 6.11: Modify Tickets**



## Chapter 7

### CONCLUSION

In conclusion, the Airlines Database Management System presented in this project serves as a comprehensive solution for managing flight information, facilitating customer bookings, and enabling efficient ticket management. By offering a user-friendly interface, customers can easily access available flights, select their desired source and destination, and proceed with booking tickets by providing essential passenger details. Additionally, the system empowers users to manage their bookings effectively by enabling ticket deletion and modification functionalities. Through its robust features and seamless functionality, the Airlines Database Management System enhances the overall experience for both customers and airline personnel, optimizing efficiency and convenience in the airline industry.

## Chapter 8

### FUTURE ENHANCEMENTS

Future upgrades to this project will implement:

- 1.Seat Selection: Integrate a seat selection feature that allows customers to choose their preferred seats during the booking process, offering them greater control and customization.
- 2.Payment Gateway Integration: Incorporate secure payment gateways to facilitate online payments for ticket bookings, supporting various payment methods such as credit/debit cards, digital wallets, and net banking.
- 3.Flight Status Updates: Integrate real-time flight status updates to keep customers informed about any changes or delays in their scheduled flights, enhancing transparency and reducing inconvenience.
- 4.Feedback and Rating System: Implement a feedback and rating system to collect customer feedback and ratings on their flight experiences, enabling the airline to continuously improve its services based on customer feedback.
- 5.Enhanced Security Measures: Strengthen security measures to protect customer data and transactions, incorporating features such as two-factor authentication and encryption to ensure the confidentiality and integrity of sensitive information.

By incorporating these future enhancements, the Airlines Database Management System can further enhance its functionality, usability, and overall value proposition, thereby improving customer satisfaction and loyalty while driving operational efficiency for the airline.

## REFERENCES

- [1] Ramakrishnan, R., &Gehrke, J. (2011). Database management systems. Boston: McGraw-Hill.
- [2] Monson-Haefel, R. (2007). J2EE Web services. Boston, Mass: Addison-Wesley.
- Silberschatz.,KorthH.F.,&SudarshanS.(2011).
- [3] <https://www.w3schools.com>
- [4] <https://www.canvasjs.com>
- [5] <https://getbootstrap.com>

