

**EX NO : 01**

**CREATION OF HTML PAGES WITH FRAMES, LINKS, TABLES  
AND OTHER TAGS**

**DATE :**

**AIM:**

To create a HTML web pages using frames, tables, links and other tags.

**PROCEDURE:**

1. Divide the browser window into two sections.
2. The left section displays a table containing National Symbols in the table.
3. The right section shows the text message.
4. When the link is clicked in the left section is clicked it will displays a national symbols image in the right side section .

OUTPUT:

NATIONAL SYMBOLS

National Animal	Tiger
National Bird	Peacock
National Flower	Lotus
National Fruit	Mango
National Game	Hockey

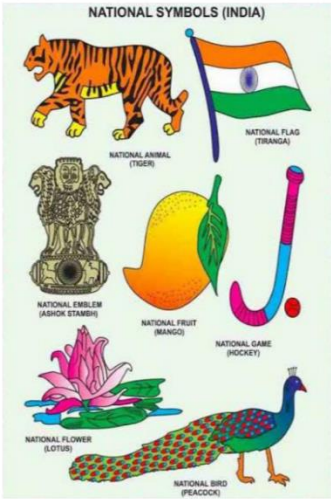
[Click here](#)

Please click the button to load the National Symbols of India...

NATIONAL SYMBOLS

National Animal	Tiger
National Bird	Peacock
National Flower	Lotus
National Fruit	Mango
National Game	Hockey

[Click here](#)



## PROGRAM:

### Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<frameset cols="60%,*">
  <frame src="frame1.html">
  <frame src="frame2.html" name="frame2">
</frameset>
</html>
```

### frame1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <h1>NATIONAL SYMBOLS</h1>
  <table border="2">
    <tr>
      <th>National Animal</th>
      <td>Tiger</td>
    </tr>
    <tr>
      <th>National Bird</th>
      <td>Peacock</td>
    </tr>
  </table>
</body>
</html>
```



```
<tr>

  <th>National Flower</th>

  <td>Lotus</td>

</tr>

<tr>

  <th>National Fruit</th>

  <td>Mango</td>

</tr>

<tr>

  <th>National Game</th>

  <td>Hockey</td>

</tr>

</table>

<br>

<a href="frame3.html" target="frame2">Click here</a>

</body>

</html>
```

## frame2.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document</title>

</head>

<body>

  <h1>Please click the button to load the National Symbols of India...</h1>

</body>

</html>
```

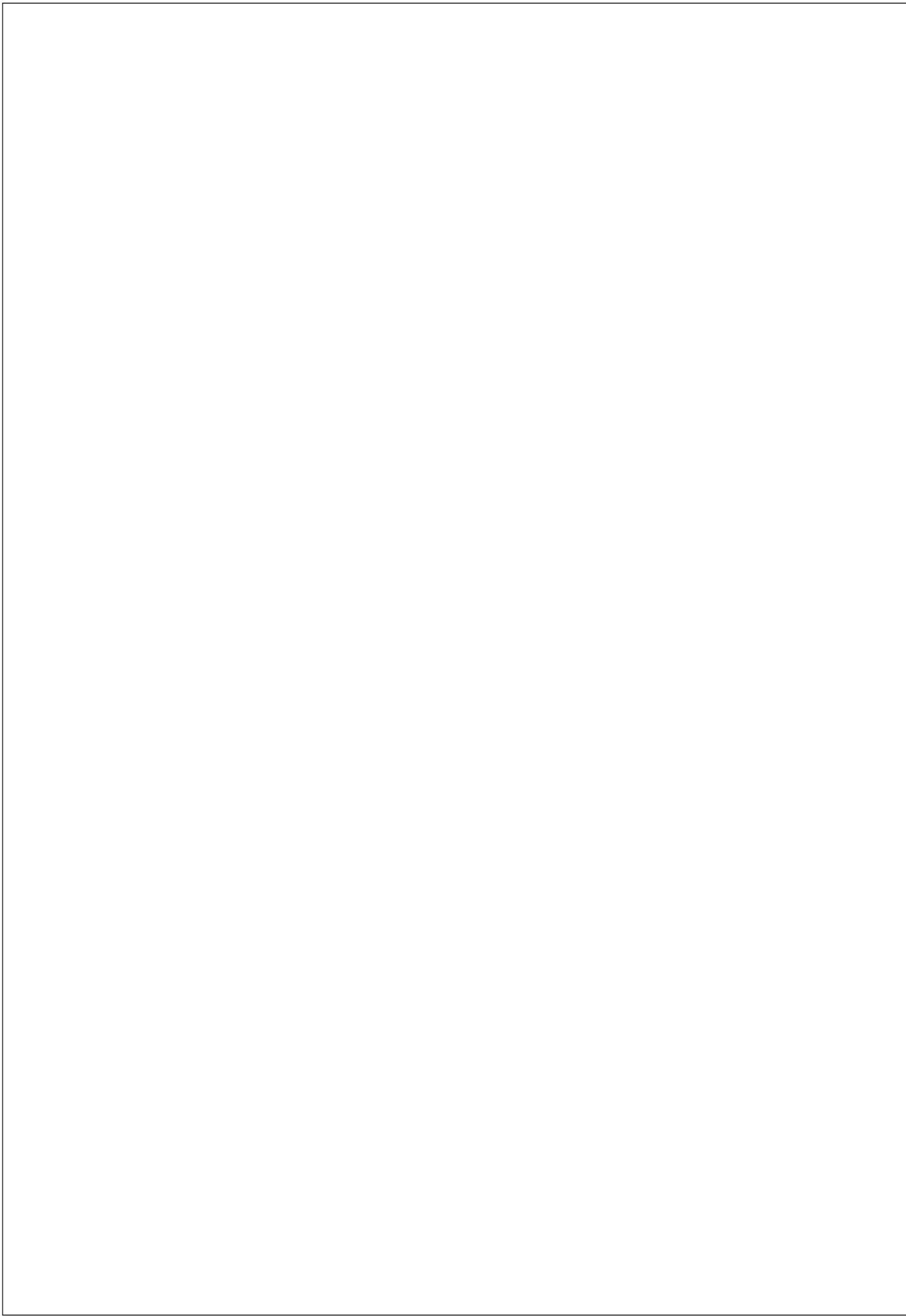


### **frame3.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  
</body>
</html>
```

### **RESULT:**

Thus, the program to create a simple web page using frames, lists, tables are coded and output is verified successfully.





**AIM:**

To create a web page with the following using HTML.

1. To embed an image map in web page.
2. To fix the hotspots.
3. Show all the related information when it is clicked.

**PROCEDURE:****1. Create the Basic HTML Structure:**

- Set up the web page with the basic HTML elements such as <html>, <head>, and <body>. Add a suitable <title> for the page.

**2. Insert the Image Map:**

- Use the <img> tag to insert an image into the webpage, and link it to an image map using the usemap attribute.

**3. Define Hot Spots:**

- Inside the <map> tag, use the <area> tag to define hot spots on the image. Each <area> should have a coords attribute to specify the clickable region and an href that links to different sections on the same page.

**4. Create Information Sections:**

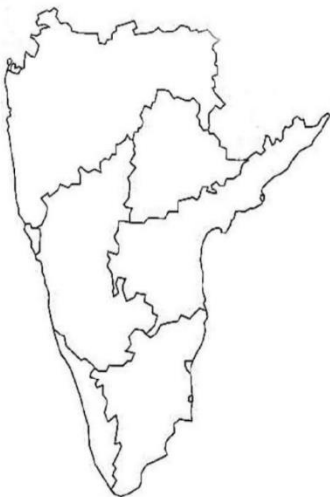
- For each hot spot, create corresponding <div> sections with unique IDs that match the href of the hot spots. These sections will contain the information displayed when the hot spot is clicked.

**5. Test the Functionality:**

- Click on the hot spots to ensure that the correct information is displayed when each hot spot is selected.

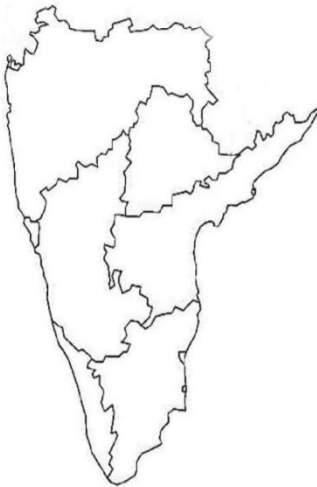
**OUTPUT:**

**South India Map**



Click on a part to see its features.

**South India Map**



Tamil Nadu is a state in southern India. It covers more than 50,200 square miles (130,000 square km). Tamil people constitute the majority of the state population, and Tamil is the state official language. Tamil Nadu capital is Chennai.

## PROGRAM:

### index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Interactive South India Map</title>

  <style>

    #map-container {

      text-align: center;

    }

    #description {

      margin-top: 20px;

      font-size: 18px;

      text-align: center;

    }

  </style>

</head>

<body>

  <h1 style="text-align: center;">South India Map</h1>

  <div id="map-container">

    <map name="South India-map">

      <area shape="poly" coords="19,92,43,265,232,115,320,125,303,41,73,20" href="#" alt="Maharashtra"

onclick="showDescription('maharashtra')">

      <area shape="circle" coords="217,467,49" href="#" alt="TamilNadu" onclick="showDescription('TamilNadu')">

      <area shape="circle" coords="76,507,102" href="#" alt="Kerala" onclick="showDescription('Kerala')">

      <area shape="poly" coords="172,263,226,367,297,364,297,274,467,147" href="#" alt="Andhra Pradesh"

onclick="showDescription('Andhra Pradesh')">

      <area shape="circle" coords="255,188,62" href="#" alt="Telangana" onclick="showDescription('Telangana')">

      <area shape="rect" coords="68,235,180,418" href="#" alt="Karnataka" onclick="showDescription('Karnataka')">

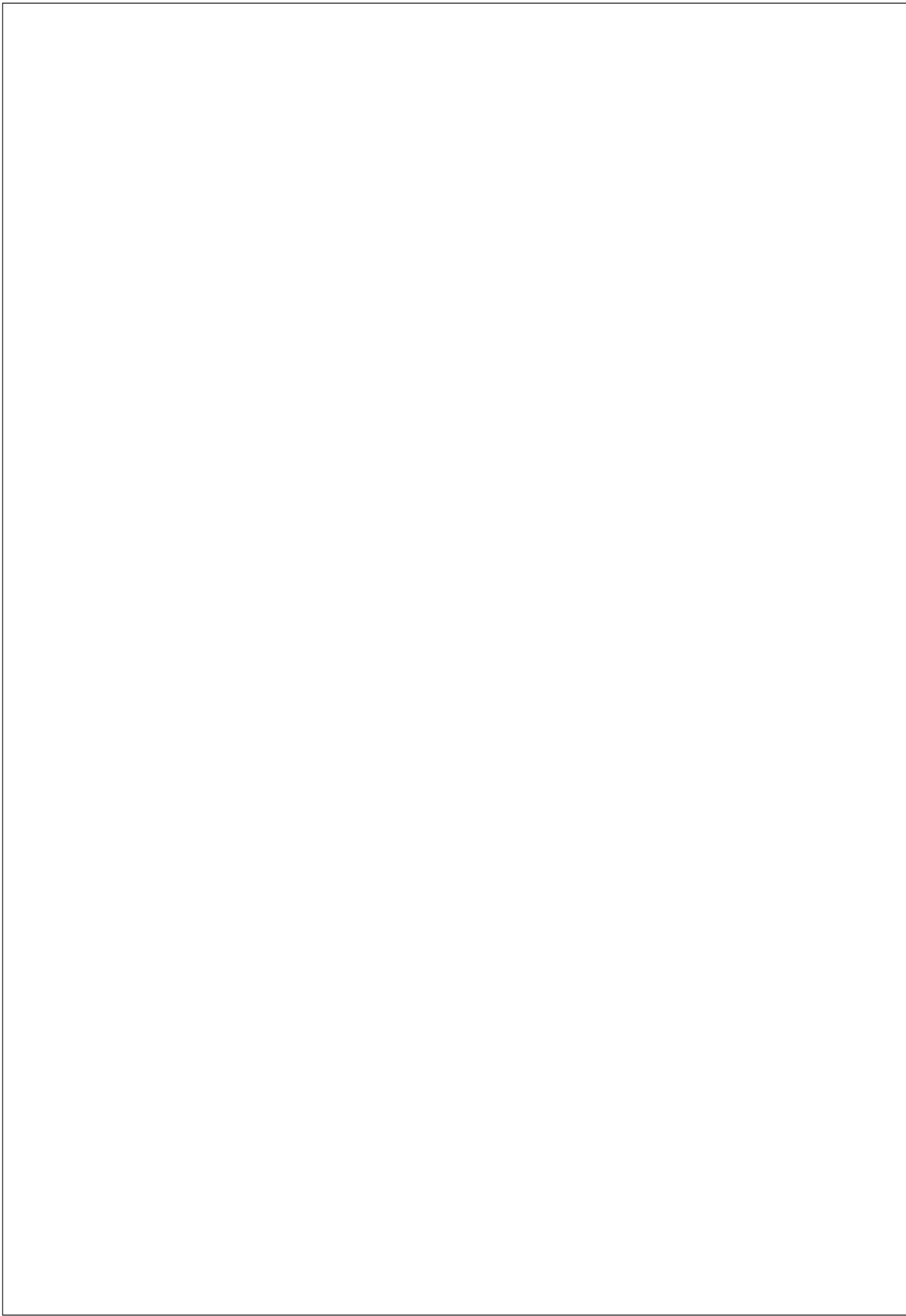
    </map>

  </div>

  <div id="description">Click on a part to see its features.</div>

  <script>

    function showDescription(continent) {
```



```

let description = "";

switch(continent) {
    case 'maharashtra':
        description = "Maharashtra, state of India, occupying a substantial portion of the Deccan plateau in the
western peninsular part of the subcontinent.";
        break;
    case 'TamilNadu':
        description = "Tamil Nadu is a state in southern India. It covers more than 50,200 square miles (130,000
square km). Tamil people constitute the majority of the state population, and Tamil is the state official language.
Tamil Nadu capital is Chennai.";
        break;
    case 'Kerala':
        description = "Kerala, southwestern coastal state of India. It is a small state, constituting only about 1
percent of the total area of the country.";
        break;
    case 'Andhra Pradesh':
        description = "Andhra Pradesh is a state in southeastern India, known for its rich cultural heritage, Telugu
language, and vibrant traditions.";
        break;
    case 'Telangana':
        description = "Telangana is a state in southern India, formed in 2014, known for its rich history, the Telugu-
speaking population, and the vibrant city of Hyderabad.";
        break;
    case 'Karnataka':
        description = "Karnataka is a state in southwestern India, known for its rich history, diverse cultures, and
vibrant IT industry, especially in Bengaluru. ";
        break;
    default:
        description = "Click on a continent to see its features.";
}

document.getElementById('description').innerText = description;
}
</script>
</body>
</html>

```

## RESULT:

Thus, the program for hotspot creation is successfully created.



**AIM:**

To create a web page using internal and external CSS along with the HTML pages.

**PROCEDURE:****1. Create the Main Page (index.html):**

- Define a <navbar> element, using the unordered list to create a navigation bar of the restaurant page contains fields like home, contact, about and login.

**2. Design a welcome message(index.html):**

- Create a welcome message using heading tag and adding styles by using external CSS.

**3. Design Button (index.html):**

- Create a button that navigate to the menu page and adding styles for it through inline CSS.

**4. Create the external CSS (INDEX.css):**

- By using this we add styles to the contents in the index.html page.

**5. Link and Test:**

- Ensure proper linking with INDEX.css and test the functionality in a browser.

OUTPUT:





## PROGRAM:

### index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link rel="stylesheet" href="INDEX.css">
  <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>
  <style>
body
{
  background-image: url('https://wallpaperaccess.com/full/3014720.jpg');
  background-repeat: no-repeat;
  background-size: cover;
}

</style>
</head>
<body>
  <div class="dv" >
    <navbar class="NAV">
      <div>
        <ul class="nav1">
          <li class="r1"><i class='bx bxs-bowl-hot' ></i>Resto.</li>
        </ul>
        <ul class="left">

          <li class="right">Home</li>
          <li class="right">Menu</li>
          <li class="right">About</li>
          <li class="right">Contact</li>
          <li class="right">Login</li>
        </ul>
      </div>
    </navbar>
  </div>
  <p style="color: aliceblue;font-size: 100px;margin-left: 30%;margin-top: 5%;font-weight: bold;font-family:monospace">Welcome to Resto.</p>
```



```
<button class="get">
  See Menu
</button>
</body>
</html>
```

## INDEX.css

```
*{

  margin: 0;

}

.get{

  color: rgb(249, 248, 247);

  margin-left: 42%;

  font-size: 50px;

  padding:10px;

  background-color: rgba(255, 99, 71, 0.181);

  font-weight: 600;

  margin-top:5%;

  border-radius: 15px;

  border:2px solid white;

}

.nav1

{

  height:50px;

  margin-top:20px;

}

.r1{

  margin-top: 0;;

  list-style: none;

  font-size: 50px;
```



```
color: rgb(248, 250, 252);

font-family: monospace;

font-weight:800;

padding-top: 22px;

}

.left{

    margin-top:-25px;

    display: flex;

    justify-content: end;

}

.right

{margin-top: 0;

    margin-right: 40px;

    list-style: none;

    font-size: 30px;

color: rgba(248, 242, 242, 0.999);

font-family: monospace;

font-weight:800;

padding-bottom: 20px;

}
```

## **RESULT:**

Thus, the program to create a simple web page using internal and external CSS with HTML pages are coded and output is verified successfully.



**AIM:**

To create a form validation containing textfield, radio buttons, checkboxes, listbox and other controls.

**PROCEDURE:****1. Create the Main Page (index.html):**

- Create a main page containing textfield, radio buttons, checkboxes, listboxes and other controls and link the style.css with it.

**2. Validate name and email:**

- Check if the namefield contains only letters and spaces using nameRegex.
- Check if the email field matches the standard email format using emailRegex.
- If anything is invalid then, display an error message and prevent form submission.

**3. Validate Gender in Radio buttons:**

- Check if any gender radio button is selected, if not then display an error message and prevent form validation.

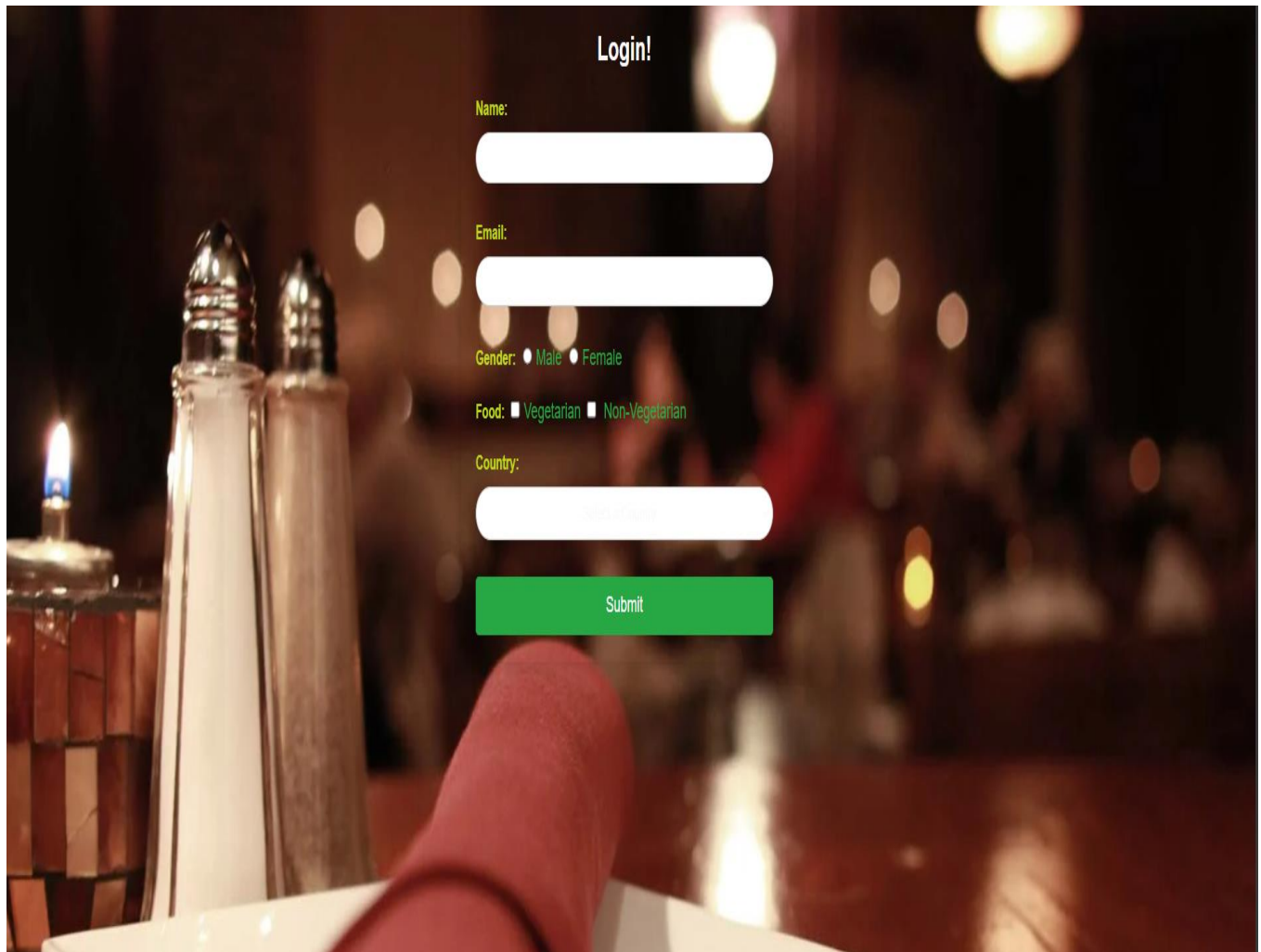
**4. Validate Gender in Checkbox:**

- Check if any food preference checkbox is selected, if not then display an error message and prevent form validation.

**5. Submit:**

- If all validations pass, allow the form to submit.

## OUTPUT:



**Login!**

Name:

Email:

Gender: ☒ Male ☐ Female

Food: ☐ Vegetarian ☐ Non-Vegetarian

Country:



## PROGRAM:

### index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Login</title>

  <link rel="stylesheet" href="style.css">

</head>

<body><form>

  <h2>Login!</h2>

  <label for="name">Name:</label>

  <input type="text" id="name"><br><br>

  <label for="Email">Email:</label>

  <input type="email" id="Email"><br><br>

  <label for="gender">Gender:</label>

  <input type="radio" name="gender" id="no" value="Male"><span id="no">Male</span>

  <input type="radio" name="gender" id="no" value="Female"><span id="no">Female</span>

  <br><br>

  <label for="food">Food:      </label>

  <input type="checkbox" name="food" id="no" value="vegetarian" /><span id="no">Vegetarian</span>

  <br>

  <input type="checkbox" name="food" id="no" value="non-vegetarian" /> <span id="no">Non-Vegetarian</span>

  <br><br>

  <label for="country">Country:</label>

  <select id="country">

    <option value="">-Select a Country-</option>

    <option value="USA">USA</option>

    <option value="Canada">Canada</option>
```



```
<option value="India">India</option>

</select><br><br>

<input type="submit">

</form>

<script>
    document.getElementById('myForm').onsubmit = function(event) {

var name = document.getElementById('name').value;

var email = document.getElementById('email').value;

var genders = document.getElementsByName('gender');

var foodPreferences = document.getElementsByName('food');

var country = document.getElementById('country').value;

var nameRegex = /^[a-zA-Z\s]+$/; // Allows only alphabets and spaces
var emailRegex = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;

if (!nameRegex.test(name)) {

    alert('Name can only contain letters and spaces');

    event.preventDefault();

}

if (!emailRegex.test(email)) {

    alert('Please enter a valid email address');

    event.preventDefault();

}

var isGenderSelected = false;

for (var i = 0; i < genders.length; i++) {

    if (genders[i].checked) {

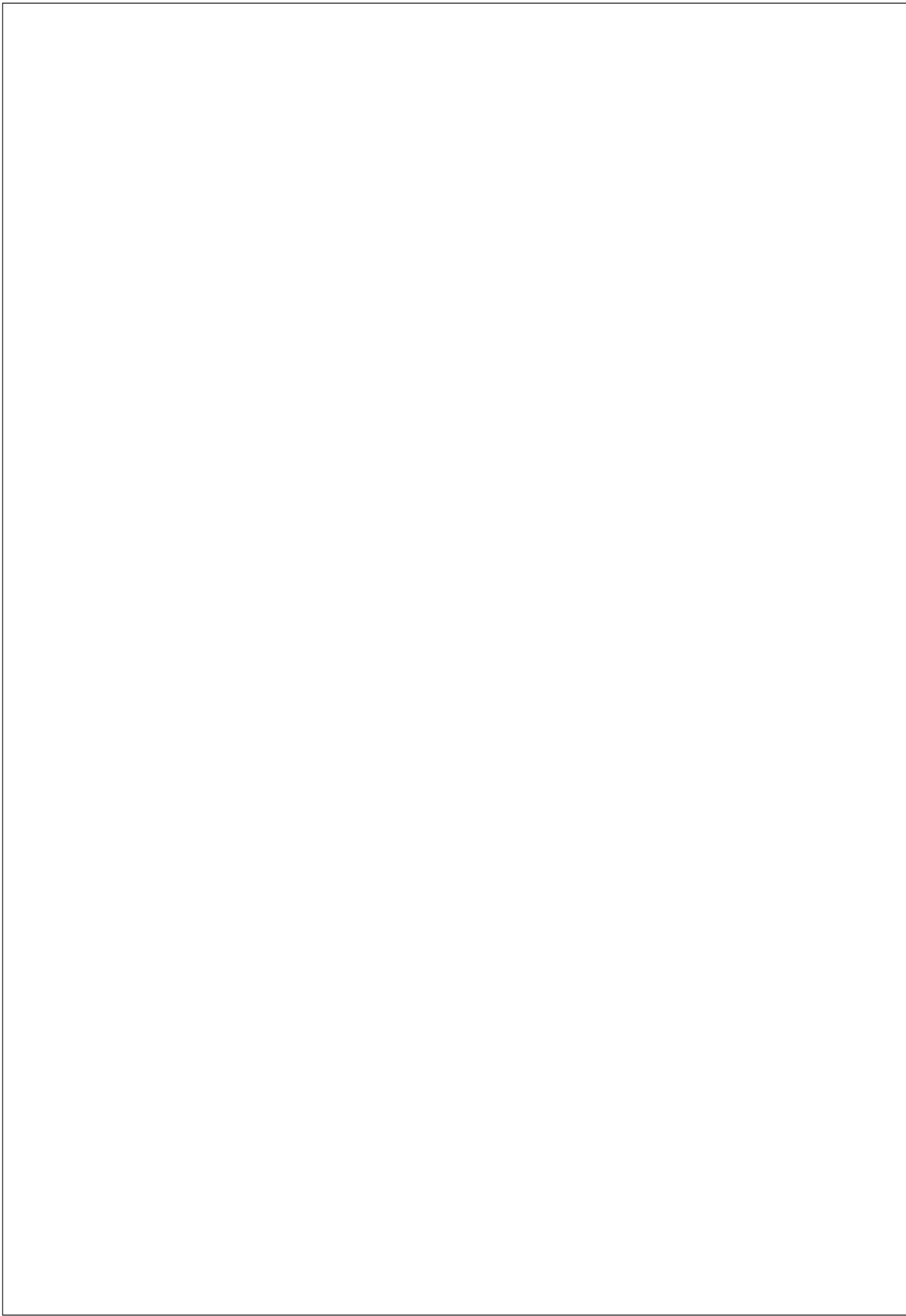
        isGenderSelected = true;

        break;

    }

}

}
```



```
if (!isGenderSelected) {  
    alert('Please select your gender');  
    event.preventDefault();  
}  
  
var isFoodChecked = false;  
for (var i = 0; i < foodPreferences.length; i++) {  
    if (foodPreferences[i].checked) {  
        isFoodChecked = true;  
        break;  
    }  
}  
  
if (!isFoodChecked) {  
    alert('Please select at least one food preference');  
    event.preventDefault();  
}  
  
if (country === "") {  
    alert('Please select a country');  
    event.preventDefault();  
}  
};  
</script>  
</body>  
</html>
```



## Style.css

```
body {
    font-family: Arial, sans-serif;
    background-image: url('https://wallpaperaccess.com/full/3014720.jpg');
    background-repeat: no-repeat;
    background-size: cover;
    margin: 0;
    padding: 20px;
    color: #333;
    margin-top: 8%;
}

form {
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
    width: 400px;
    margin:auto;
    margin-top:-10%;
}

h2 {
    text-align: center;
    color: #fdf9f9;
    font-size: 24px;
    margin-bottom: 20px;
}

label {
    font-size: 14px;
    font-weight: bold;
    color:#bed818;
}
```





```
input[type="text"],
input[type="email"],
select {
    width: 100%;
    padding: 10px;
    margin: 8px 0;
    color: #fdf9f9;
    box-sizing: border-box;
    border: 1px solid #ccc;
    border-radius: 20px;
}
```

```
input[type="radio"],
input[type="checkbox"] {
    margin-right: 5px;
```

```

}
#no{
color:#28a745;
}
```

```
.gender-group, .food-group {
    margin-bottom: 15px;
}
```

```
input[type="submit"] {
    background-color: #28a745;
    color: white;
    padding: 12px 20px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    width: 100%;
    font-size: 16px;
}
```



```
input[type="submit"]:hover {  
    background-color: #218838;  
}  
#country{  
text-align: center;  
}
```

### **RESULT:**

Thus, the program to form validation including textfield, checkbox, radiobuttons, listboxes are coded and output is verified successfully.





**EX NO : 05**

## **BUS TICKET RESERVATION SYSTEM USING JSP**

**DATE :**

### **AIM:**

To write a JSP Program for Bus Ticket Reservation System

### **PROCEDURE:**

Step 1: Create a MySQL database named “bus\_reservation\_system” and a “r” table.

Step 2: Design a JSP form (bookBus.jsp) to collect user travel destination and date.

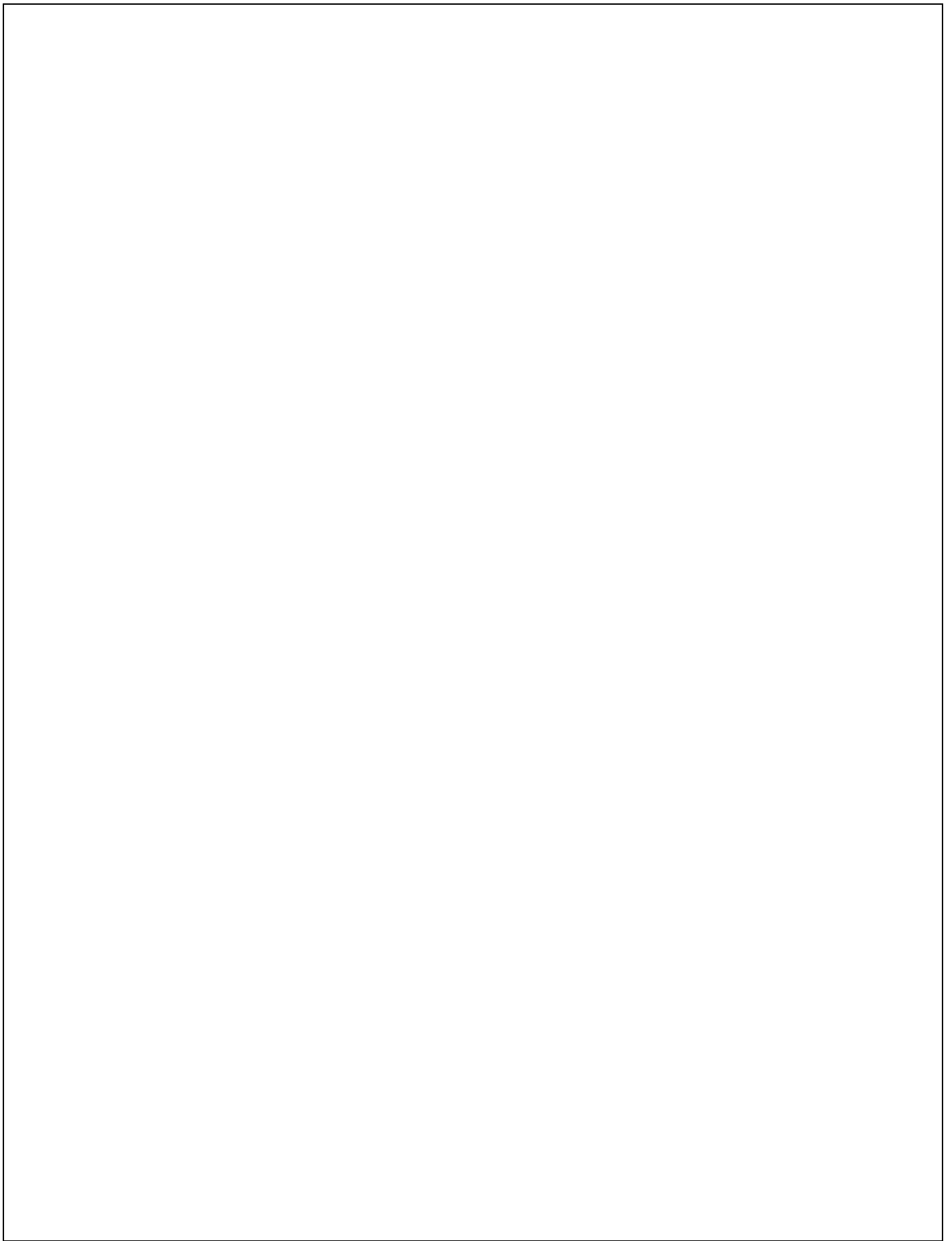
Step 3: Design a JSP form (searchBuses.jsp) to show the available seats.

Step 4: Develop a JSP page (bookingConfirmation.jsp) to retrieve form data and connect to the database.

Step 5: Develop a JSP page (confirmBooking.jsp) to confirm the booking and update the database.

Step 6: Display booking confirmation details on confirmBooking.jsp after successful insertion.

Step 7: Implement error handling and close the database connection in confirmBooking.jsp.



## PROGRAM:

### index.jsp

```
<% @ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>Bus Reservation System</title>
</head>
<body>
    <h1>Search for Buses</h1>
    <form action="searchBuses.jsp" method="get">
        <label for="route">Route:</label>
        <input type="text" name="route" id="route" required><br>
        <label for="date">Date:</label>
        <input type="date" name="date" id="date" required><br>
        <input type="submit" value="Search">
    </form>
</body>
</html>
```

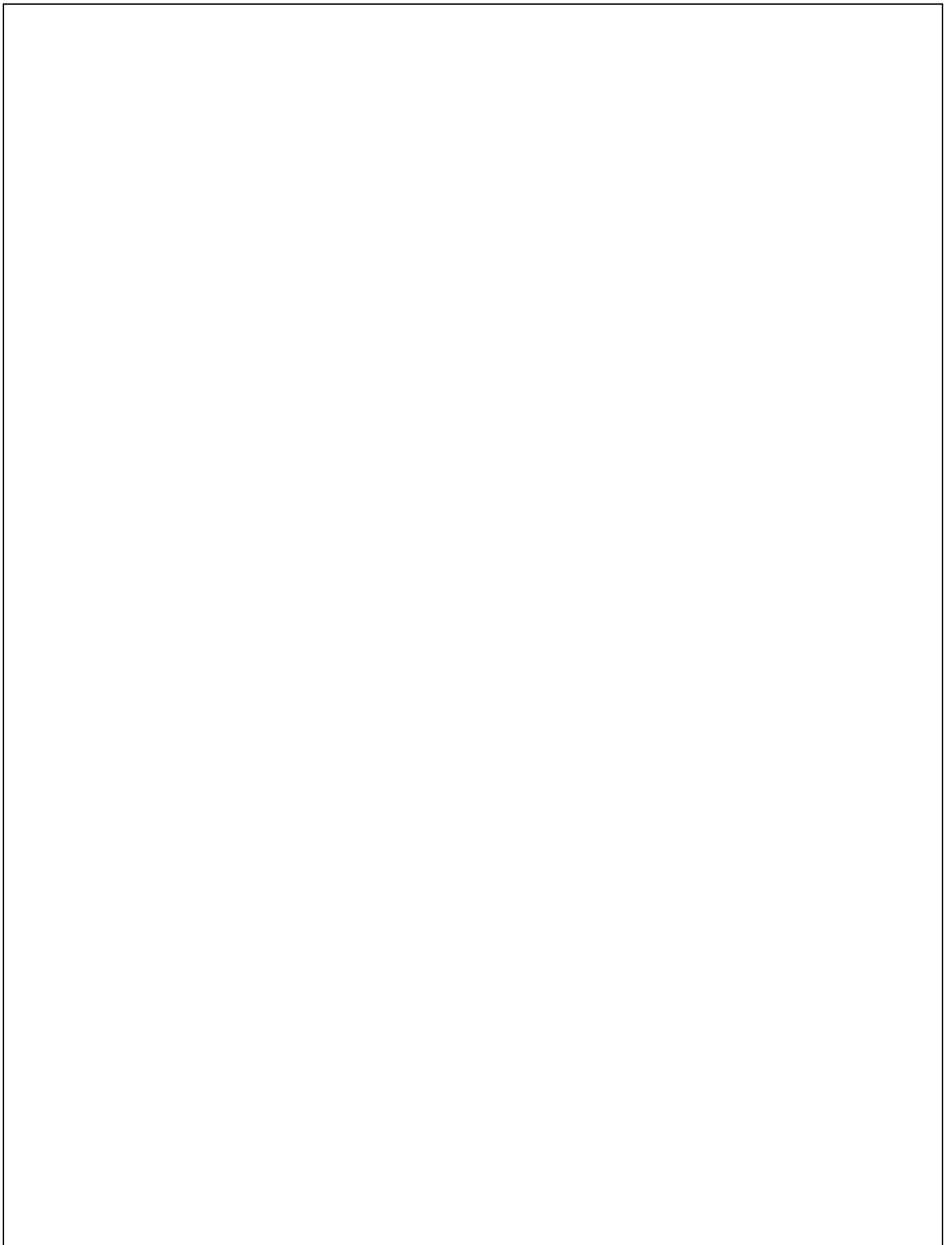
### bookBus.jsp

```
<% @ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-
8"%>
<% @ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
    <title>Book Bus</title>
</head>
<body>
    <h1>Book Bus</h1>

    <%
        int busId = Integer.parseInt(request.getParameter("bus_id"));

        Connection conn = null;
        PreparedStatement pstmt = null;
        ResultSet rs = null;
```





```

try {
    // Database connection
    Class.forName("com.mysql.cj.jdbc.Driver");
    conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/bus_reservation_system",
    "root", "root");

    // Query to get bus details
    String sql = "SELECT * FROM buses WHERE bus_id = ?";
    pstmt = conn.prepareStatement(sql);
    pstmt.setInt(1, busId);
    rs = pstmt.executeQuery();

    if (rs.next()) {
        %>
        <h2>Bus ID: <%= rs.getInt("bus_id") %></h2>
        <p>Route: <%= rs.getString("route") %></p>
        <p>Travel Date: <%= rs.getDate("travel_date") %></p>
        <p>Available Seats: <%= rs.getInt("available_seats") %></p>

        <form action="confirmBooking.jsp" method="post">
            <input type="hidden" name="bus_id" value="<%= rs.getInt("bus_id") %>">
            <label for="num_seats">Number of Seats:</label>
            <input type="number" name="num_seats" id="num_seats" required min="1"
max="<%= rs.getInt("available_seats") %>">
            <input type="submit" value="Confirm Booking">
        </form>

        <%
        } else {
        %>
        <p>Bus not found.</p>
        <%
        }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (rs != null) try { rs.close(); } catch (SQLException e) { e.printStackTrace(); }
            if (pstmt != null) try { pstmt.close(); } catch (SQLException e) {
e.printStackTrace(); }
            if (conn != null) try { conn.close(); } catch (SQLException e) { e.printStackTrace(); }
            }
        %>
    </body>
</html>

```

# OUTPUT:

```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 78
Server version: 8.0.34 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use bus_reservation_system;
Database changed
mysql> select * from buses;
+-----+-----+-----+-----+
| bus_id | route           | travel_date | available_seats |
+-----+-----+-----+-----+
| 1      | City A - City B | 2024-10-20  | 50               |
| 2      | City C - City D | 2024-10-21  | 50               |
| 3      | City A - City D | 2024-10-22  | 50               |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> |
```

← → ↺

localhost:8082/bus1/

☆ 📌 🔔 ⋮

📱

Gmail

SSO Portal

YouTube

Maps

Imported From IE

https://tableau.mahil...

Search for Buses

Route: City A - City B

Date: 20-10-2024 📅

Search

## searchBuses.jsp

```
<% @ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<% @ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
    <title>Available Buses</title>
</head>
<body>
    <h1>Available Buses</h1>

    <%
        String route = request.getParameter("route");
        String date = request.getParameter("date");

        Connection conn = null;
        PreparedStatement pstmt = null;
        ResultSet rs = null;

        try {
            // Database connection
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/bus_reservation_system",
"root", "root");

            // Log input values
            System.out.println("Route: " + route);
            System.out.println("Travel Date: " + date);

            // Modified query to only filter by date
            String sql = "SELECT * FROM buses WHERE travel_date = ?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setDate(1, Date.valueOf(date));
            rs = pstmt.executeQuery();
        }
    %>

    <table border="1">
        <tr>
            <th>Bus ID</th>
            <th>Route</th>
            <th>Travel Date</th>
            <th>Available Seats</th>
            <th>Action</th>
        </tr>
```

# Available Buses

Bus ID	Route	Travel Date	Available Seats	Action
1	City A - City B	2024-10-20	50	<a href="#">Book Now</a>

# Book Bus

Bus ID: 1

Route: City A - City B

Travel Date: 2024-10-20

Available Seats: 50

Number of Seats:

Confirm Booking

```

<%
    boolean hasBuses = false;
    while (rs.next()) {
        hasBuses = true;
    }

    <tr>
        <td><%= rs.getInt("bus_id") %></td>
        <td><%= rs.getString("route") %></td>
        <td><%= rs.getDate("travel_date") %></td>
        <td><%= rs.getInt("available_seats") %></td>
        <td><a href="bookBus.jsp?bus_id=<%= rs.getInt("bus_id") %>">Book
Now</a></td>
    </tr>
<%
    }

    if (!hasBuses) {
    <tr>
        <td colspan="5">No buses available for the selected date.</td>
    </tr>
    <%
        }
    <%
    </table>

    <%
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (rs != null) try { rs.close(); } catch (SQLException e) { e.printStackTrace(); }
        if (pstmt != null) try { pstmt.close(); } catch (SQLException e) { e.printStackTrace(); }
    }
    if (conn != null) try { conn.close(); } catch (SQLException e) { e.printStackTrace(); }
    }
    <%
</body>
</html>

```

### **confirmBooking.jsp**

```

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
    <title>Booking Confirmation</title>
</head>

```

# Booking Confirmation

Your booking is confirmed! You have booked 23 seats.

```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 78
Server version: 8.0.34 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use bus_reservation_system;
Database changed
mysql> select * from buses;
+-----+-----+-----+-----+
| bus_id | route          | travel_date | available_seats |
+-----+-----+-----+-----+
| 1 | City A - City B | 2024-10-20 | 50 |
| 2 | City C - City D | 2024-10-21 | 50 |
| 3 | City A - City D | 2024-10-22 | 50 |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> select * from buses
-> ;
+-----+-----+-----+-----+
| bus_id | route          | travel_date | available_seats |
+-----+-----+-----+-----+
| 1 | City A - City B | 2024-10-20 | 27 |
| 2 | City C - City D | 2024-10-21 | 50 |
| 3 | City A - City D | 2024-10-22 | 50 |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> |
```

```

<body>
  <h1>Booking Confirmation</h1>
  <%
    int busId = Integer.parseInt(request.getParameter("bus_id"));
    int numSeats = Integer.parseInt(request.getParameter("num_seats"));
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
      // Database connection
      Class.forName("com.mysql.cj.jdbc.Driver");
      conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/bus_reservation_system",
"root", "root");
      // Update available seats
      String sql = "UPDATE buses SET available_seats = available_seats - ? WHERE
bus_id = ?";
      pstmt = conn.prepareStatement(sql);
      pstmt.setInt(1, numSeats);
      pstmt.setInt(2, busId);
      int rowsAffected = pstmt.executeUpdate();
      if (rowsAffected > 0) {
        %>
          <p>Your booking is confirmed! You have booked <%= numSeats %> seats.</p>
        <%
          } else {
        %>
          <p>Booking failed. Please try again.</p>
        <%
          }
      } catch (Exception e) {
        e.printStackTrace();
      } finally {

```





```
        if (pstmt != null) try { pstmt.close(); } catch (SQLException e) { e.printStackTrace(); }  
        if (conn != null) try { conn.close(); } catch (SQLException e) { e.printStackTrace(); }  
    }  
    %>  
</body>  
</html>
```

**RESULT:**

Thus the program using jsp for bus ticket reservation system was created successfully and the output has been verified



**EX NO : 07**

## **XML DOCUMENT**

**DATE :**

### **AIM:**

To create a XML document for representing the Student details .

### **PROCEDURE:**

**Step 1:** Define the XML version and encoding:

- `<?xml version="1.0" encoding="UTF-8"?>`

**Step 2:** Create the root element `<StudentDetails>`.

**Step 3:** Add a `<student>` element for each student, including:

- `<firstname>` with the student's first name.
- `<lastname>` with the student's last name.
- `<age>` with the student's age.

**Step 4:** Close the root element `<StudentDetails>`.

**Step 5:** Save the XML document.

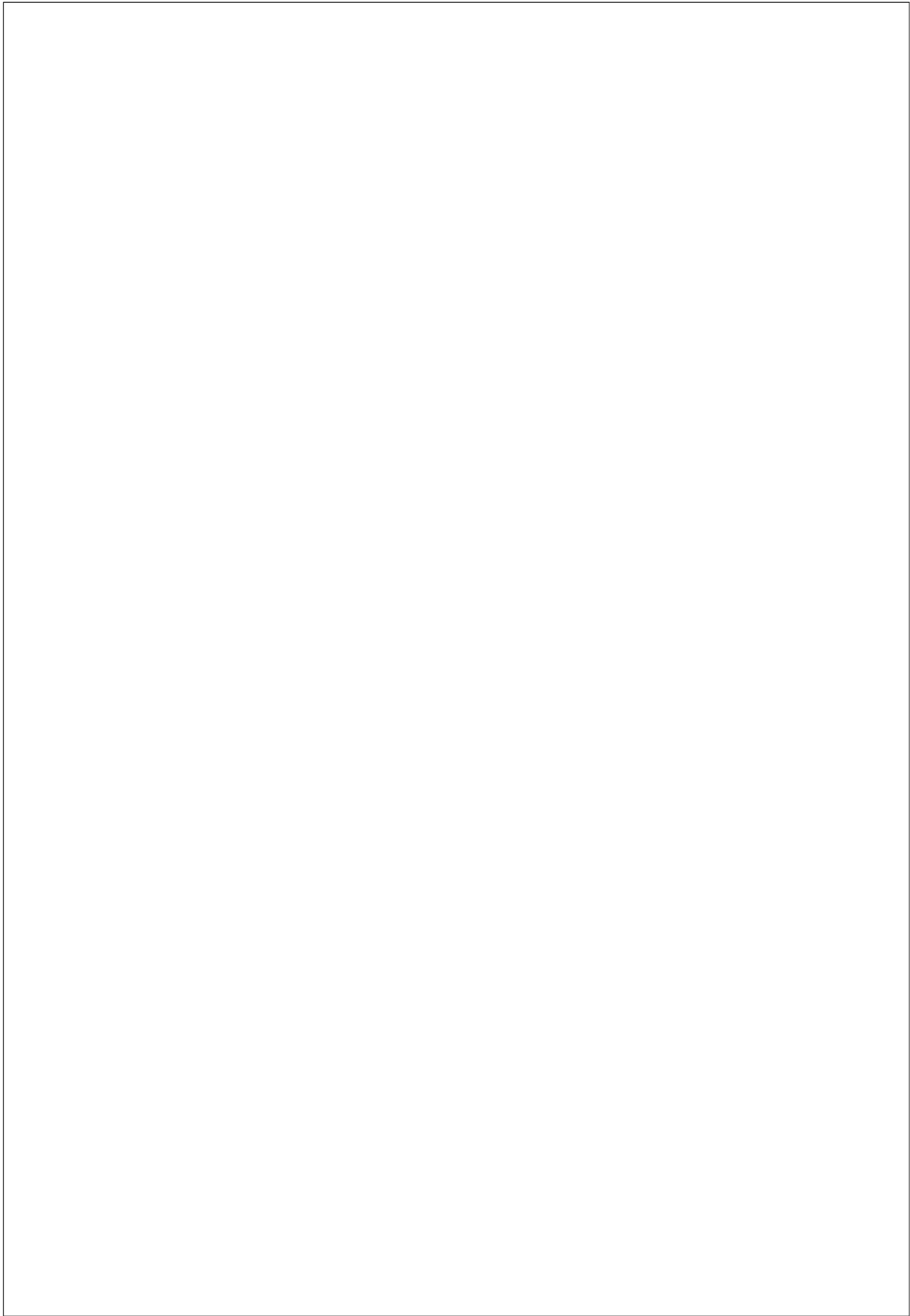


**PROGRAM:**

```
<?xml version="1.0" encoding="UTF-8"?>
<StudentDetails>
  <student>
    <firstname>Lejesh</firstname>
    <lastname>kannan</lastname>
    <age>20</age>
  </student>
  <student>
    <firstname>Keerthi</firstname>
    <lastname>kannan</lastname>
    <age>15</age>
  </student>
  <student>
    <firstname>Jack</firstname>
    <lastname>son</lastname>
    <age>10</age>
  </student>
</StudentDetails>
```

**RESULT:**

Thus , the XML Document for the students details was created successfully and the output has been verified.



**EX NO : 08**

## **XML SCHEMA**

**DATE :**

### **AIM:**

To create a XML schema for representing the Student details .

### **PROCEDURE:**

**Step 1:** Define the XML schema version and namespace.

- `<?xml version="1.0" encoding="UTF-8"?>`
- `xmlns:xs="http://www.w3.org/2001/XMLSchema"` specifies the XML Schema namespace.

**Step 2:** Create the root element `<xs:schema>` to define the schema.

**Step 3:** Define the main element `<StudentDetails>` using `<xs:element name="StudentDetails">`.

**Step 4:** Inside `<StudentDetails>`, define a complex type to allow a sequence of child elements.

**Step 5:** Add a `<student>` element that can appear multiple times (`maxOccurs="unbounded"`):

- This allows any number of student records to be included.

**Step 6:** Within `<student>`, specify a sequence of elements:

- **`<firstname>`:** Must be a string (`xs:string`).
- **`<lastname>`:** Must be a string (`xs:string`).
- **`<age>`:** Must be an integer (`xs:int`).

**Step 7:** Close all tags properly to complete the schema.



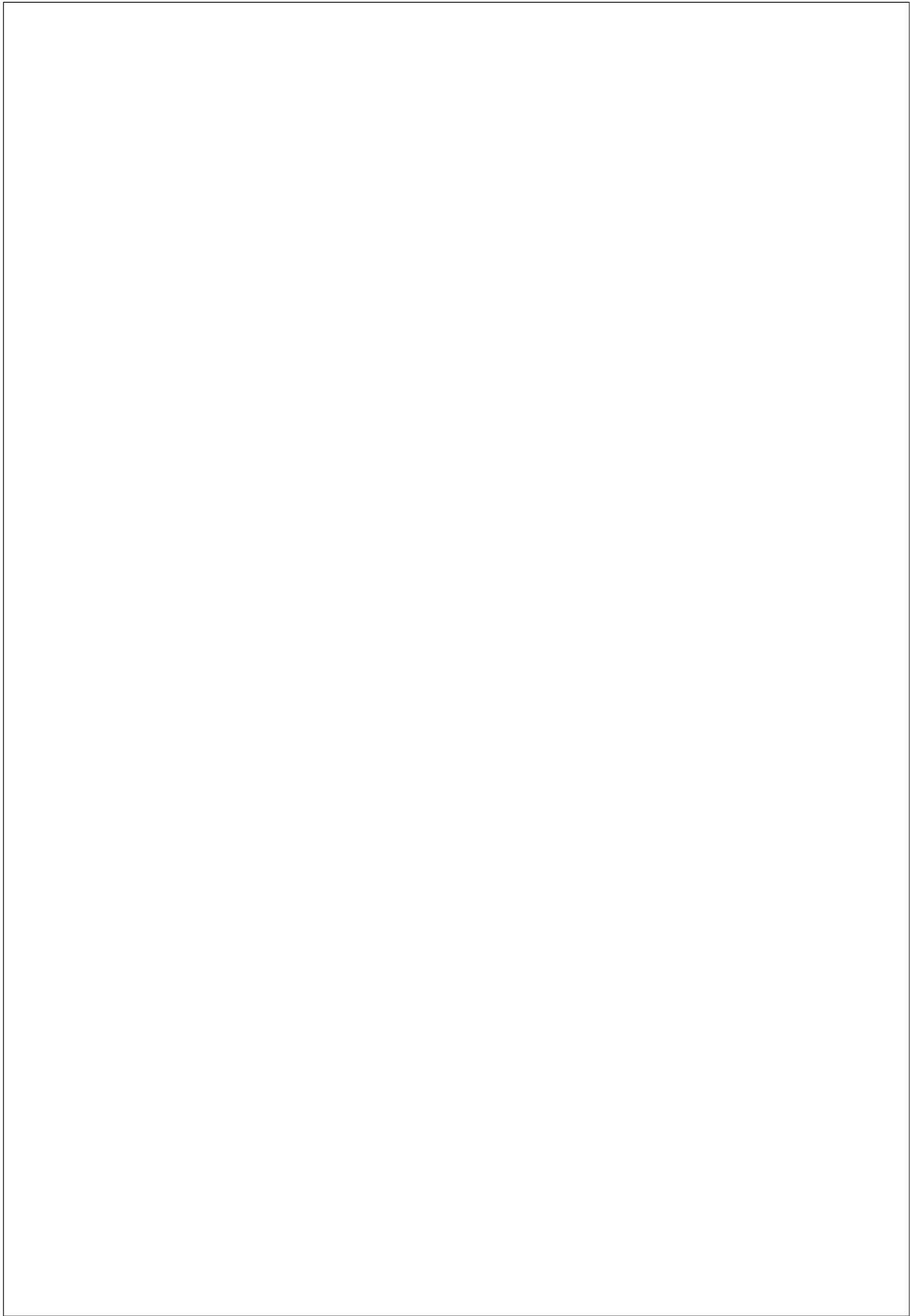


**PROGRAM:**

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="StudentDetails">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="student" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="firstname" type="xs:string" />
              <xs:element name="lastname" type="xs:string" />
              <xs:element name="age" type="xs:int" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

**RESULT:**

Thus, the XML Schema for student details was created successfully and the output has been verified.



**EX NO : 09**

## **PARSER XML DOCUMENT USING DOM PARSER**

**DATE :**

### **AIM:**

To write a java program to read the data from the XML document using DOM parser.

### **PROCEDURE:**

**Step 1:** Import required libraries for XML parsing and file handling.

**Step 2:** Create a File object for the XML file (students.xml).

**Step 3:** Use DocumentBuilderFactory to create a DocumentBuilder.

**Step 4:** Parse the XML file to obtain a Document object.

**Step 5:** Normalize the XML document structure.

**Step 6:** Get the root element and print its name.

**Step 7:** Retrieve all <student> elements using getElementByTagName.

**Step 8:** Iterate over each <student> node.

**Step 9:** Convert each node to an Element.

**Step 10:** Extract <firstname>, <lastname>, and <age> values.

**Step 11:** Print the extracted student details.

**Step 12:** Catch and handle any exceptions that may occur.



**PROGRAM:**

```
import org.w3c.dom.*;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import java.io.File;

public class Parser {
    public static void main(String[] args) {
        try {
            File xmlFile = new File("students.xml");
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
            Element root= doc.getDocumentElement();
            System.out.println("Root element: " + root.getNodeName());
            NodeList nodeList = root.getChildNodes();
            // Iterate through each <student> node
            for (int i = 0; i < nodeList.getLength(); i++) {
                Node node = nodeList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element element = (Element) node;
                    // Retrieve and print the data
                    String firstName = element.getElementsByTagName("firstname")
                        .item(0).getTextContent();
                    String lastName = element.getElementsByTagName
                        ("lastname").item(0).getTextContent();
                    String age = element.getElementsByTagName
                        ("age").item(0).getTextContent();
                    System.out.println("Student:");
                    System.out.println("First Name: " + firstName);
                    System.out.println("Last Name: " + lastName);
```

## **OUTPUT:**

Root element: StudentDetails

Student:

First Name: Lejesh

Last Name: kannan

Age: 20

Student:

First Name: Keerthi

Last Name: kannan

Age: 15

Student:

First Name: Jack

Last Name: son

Age: 10

```
System.out.println("Age: " + age);  
    System.out.println();  
    }  
    }  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
    }  
}
```

**RESULT:**

Thus, the program to read data from the XML document using DOM parser was written successfully and the output has been verified.