

DecorX- Decoration Booking Management

Project Report

Submitted by:

Nakka Vivek(MSS2019085)

Submitted to:

Mr. Praneeth(MSSCoordinator)

Abstract

DecorX is an online platform where you can easily plan and book beautiful decorations for events like weddings, birthdays, and engagements. It's like a one-stop shop that connects you with talented decorators. The website looks great and works smoothly on your computer or phone. Behind the scenes, it's built using technologies like HTML, CSS, and Bootstrap for a nice design. The interactive parts are made possible by JSP and Servlets. Java is the main language making sure everything runs smoothly. When you book a decoration, it's stored and managed using SQL. The whole thing is hosted and made accessible by Apache Tomcat. Customers can choose from pre-designed themes or customize decorations to make their events special. There's a secure booking system, and decorators can showcase their work, manage bookings, and get feedback from customers. DecorX is not just a website; it's a blend of technology and creativity, making event planning easier and more enjoyable for everyone involved.

Table of Contents

ABBREVIATIONS.....	2
1.Introduction.....	3
1.1 Motivation.....	4
1.2 Problem Statement.....	4
1.3 Project Objectives.....	4
2.Software Specifications.....	5
2.1Software Requirements.....	5
2.1.1Functional Requirements.....	5
2.1.2 Non-Functional Requirements:.....	6
2.2 System Specifications.....	6
2.2.1 Hardware Specifications.....	6
2.2.2 Software Specifications.....	6
3. Proposed System Design.....	8
3.1 Proposed Methods.....	8
3.2Technology Description.....	9
4.Implementation.....	10
4.1 Code Implementation.....	10
4.2 Interfaces.....	16
Conclusion.....	21
References.....	22

ABBREVIATIONS

HTML Hypertext Markup Language

JS JavaScript

CSS Cascading Style Sheets

JDBC Java DataBase Connectivity

JSP JavaServer Page

DBMS DataBase Management System

1.Introduction

1.1 Motivation

DecorX was born from the idea that everyone, regardless of their financial situation, deserves to have beautifully decorated events. After attending many events, we noticed that people, no matter their budget, really care about making their occasions special with great decorations.

So, DecorX is here to make event decoration easy and accessible for everyone. Whether it's a wedding, birthday, or any celebration, we want to provide a platform that connects people with talented decorators and offers a variety of affordable and personalized decoration options. We believe that every event, regardless of financial background, should be a visually delightful and memorable experience. DecorX is all about making that happen!

1.2 Problem Statement

DecorX is solving the problem of making beautiful event decorations accessible to everyone, regardless of their budget. Many people want to have nicely decorated events, like weddings or birthdays, but current options can be too expensive or challenging to navigate.

Our platform is designed to be easy to use and affordable. We connect people with talented decorators and offer a variety of decoration options to suit different budgets. The goal is to make event decoration a hassle-free and enjoyable experience for everyone, ensuring that every event, no matter the financial background, can be visually stunning and memorable. DecorX is here to simplify and democratize the process of creating beautiful and personalized events for all.

1.3 Project Objectives

The key objectives for DecorX are to create a user-friendly and visually appealing platform that offers affordable and customizable decoration options for various events. The project aims to connect users with skilled decorators, streamline the booking process, and provide decorators with effective tools to manage their profiles and engagements. Additionally, DecorX prioritizes security, responsiveness across devices, and continuous improvement based on user feedback, with the ultimate goal of making beautiful event decorations accessible to everyone, regardless of their financial background.

the

2. Software Specifications

2.1 Software Requirements

2.1.1 Functional Requirements

- **User Registration and Authentication:**

Users should be able to create accounts with DecorX.

- **Secure authentication mechanisms to protect user accounts.**

- **Decorator Registration and Portfolio Management:**

Decorators must be able to register and create profiles showcasing their portfolios.

Tools for decorators to manage and update their portfolios.

- **Decoration Catalog:**

A catalog of pre-designed decoration themes and styles for users to browse.

Customization options to tailor decorations based on user preferences.

- **Booking System:**

Users should be able to check decorator availability.

A seamless booking system allowing users to select event dates and complete transactions.

- **User Dashboard:**

User accounts with dashboards for customers to track bookings, save favorite designs, and manage event details.

- **Database Management:**

Efficient management of databases to store user accounts, decorator profiles, booking details, and other relevant information.

2.1.2 Non-Functional Requirements:

- **Performance:**

Ensure the platform can handle concurrent users and deliver a responsive experience.

- **Scalability:**

Design the system to scale with an increasing number of users and decorators.

- **Reliability:**

Implement measures to ensure the platform's reliability, minimizing downtime and errors.

- **Usability:**

Design an intuitive and user-friendly interface for easy navigation and interaction.

2.2 System Specifications

2.2.1 Hardware Specifications

- No hardware specifications

2.2.2 Software Specifications

1. **Frontend Tools:**

- HTML
- CSS3 and Bootstrap
- JavaScript
- JSP

2. Backend Tools:

- Servlets
- JSP
- JDBC
- Apache Tomcat (Server)

3. Database Tools:

- Oracle SQL Database

3. Proposed System Design

3.1 Proposed Methods

➤ Architecture Overview

The system will follow a client-server architecture, where the client (web browser) interacts with the server-side components.

The server-side components will be implemented using Java Servlets for handling HTTP requests and responses, along with JavaServer Pages (JSP) for generating dynamic content.

➤ Frontend Design

HTML, CSS, Bootstrap: For building the foundational structure, styling, and ensuring a responsive design.

JavaScript (JS): To add dynamic behaviour and interactivity to the user interface.

➤ Backend Design

Java: The primary back-end programming language for its reliability, scalability, and wide adoption.

Servlets: Handling server-side logic and managing requests and responses.

JSP (JavaServer Pages): Facilitating server-side rendering for dynamic content.

➤ Database

Oracle SQL: Structuring and managing the database to store user accounts, decorator profiles, booking details, and more.

JDBC (Java Database Connectivity): Enabling Java applications to interact with the Oracle database.

3.2 Technology Description

- **HTML (Hypertext Markup Language):**

Description: HTML is the standard markup language for creating the structure of web pages. It provides a set of elements to structure content on the web.

- **CSS (Cascading Style Sheets):**

Description: CSS is used for styling the HTML elements. It controls the layout and appearance of multiple web pages simultaneously.

- **Bootstrap:**

Description: Bootstrap is a popular open-source CSS framework that simplifies the design process. It provides pre-built components and a responsive grid system, making it easier to create a consistent and visually appealing layout.

- **JavaScript (JS):**

Description: JavaScript is a scripting language that enables dynamic content on web pages. It is used for client-side scripting to add interactivity and improve the user experience.

- **Java:**

Description: Java is a versatile, object-oriented programming language known for its platform independence. It is widely used for building robust, scalable backend applications.

- **Servlets:**

Description: Servlets are Java-based programs that extend the functionality of web servers. They handle requests and generate dynamic responses for web applications.

- **JSP (JavaServer Pages):**

Description: JSP is a technology used for developing dynamic web pages. It allows embedding Java code within HTML pages to create dynamic content.

- **Database Management:**

Oracle SQL:

Description: Oracle SQL is a powerful relational database management system (RDBMS) that enables efficient storage, retrieval, and manipulation of structured data.

- **JDBC (Java Database Connectivity):**

Description: JDBC is a Java API that allows Java applications to interact with relational databases. It provides a standard interface for connecting to databases and executing SQL queries.

4.Implementation

4.1 Code Implementation

➤ Index.html

```
<nav >
    <div class="navbar">
        <label id="logo">DecorX</label>
        <a href="#home">Home</a>
        <a href="./about.html">About</a>
        <a href="./contact.html">Contact</a>
        <a href="./yourorders.jsp">Your Orders</a>
        <div class="dropdown">
            <button class="dropbtn">Designs
                <i class="fa fa-caret-down"></i>
            </button>
            <div class="dropdown-content">
                <div class="dchoice" onclick="display('Marriage')">Marriage</div>
                <div class="dchoice" onclick="display('Halidi')">Halidi</div>
                <div class="dchoice" onclick="display('Reception')">Reception</div>
                <div class="dchoice" onclick="display('Birthday')">Birthday</div>
                <div class="dchoice" onclick="display('Engagement')">Engagement</div>
                <div class="dchoice" onclick="display('others')">others</div>
            </div>
        </div>
        <div class="dropdown">
            <button class="dropbtn">
                <i class="fa-solid fa-user-large ac"></i>Account
                <i class="fa fa-caret-down"></i>
            </button>
            <div class="dropdown-content" >
                <%if(session.getAttribute("username")==null) {%>
                <div class="dchoice" onclick="login('user')"><a href="./userlogin.html">User</a></div>
                <div class="dchoice" onclick="login('admin')"><a href="./adminlogin.html">Admin</a></div>
                <%>
            </div>
        </div>
    </div>
</nav>
```

➤ login.html

```
</style>
</head>
<body>
    <div id="login">
        <div><h1>User Login</h1></div>
        <br>
        <form action="UserServlet" method="post" id="loginform">
            <div>username:<input type="text" placeholder="enter your username" name="uname" required></div>
            <br>
            <div>password:<input type="password" name="password" required></div>
            <br>
            <a href="./register.html">Register here</a>
            <br>
            <div><button type="submit">Login as User</button></div>
        </form>
    </div>
</body>
</html>
```

➤ contact.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Contact Us</title>
  <link rel="stylesheet" href="./css/contact.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
  <h1>Contact Us</h1>

  <ul>
    <li>
      <a href="https://www.facebook.com/" class="fa fa-facebook"></a> Facebook
    </li>
    <li>
      <a href="https://www.twitter.com/" class="fa fa-twitter"> </a> Twitter
    </li>
    <li>
      <a href="https://www.instagram.com/" class="fa fa-instagram"> </a> Insta
    </li>
    <li>
      <a href="https://web.whatsapp.com/" class="fa fa-whatsapp"> </a> Whatsapp
    </li>
    <li>
      <a href="https://youtube.com/" class="fa fa-youtube"> </a> youtube
    </li>
  </ul>

</body>
</html>

```

➤ about.html

```

<body>
  <h1>DecorX</h1>
  <h1>About Us</h1>
  <i>
    <p>Welcome to DecorX, your premier destination for
    creating unforgettable events through stunning decorations!
    At DecorX, we specialize in providing top-notch event decoration services,
    making your special moments truly exceptional.
    </p>

    <p>Our passion is transforming ordinary spaces into extraordinary experiences.
    Whether you're planning a wedding, corporate event, birthday party,
    or any special occasion, we're here to turn your vision into reality.
    Our dedicated team of skilled decorators is committed to bringing creativity,
    style, and elegance to every event we touch.
    </p>

    <p>What sets DecorX apart is our attention to detail
    and commitment to customer satisfaction.
    We understand that each event is unique, and we work closely with our clients
    to capture their individual style and preferences.
    From concept to execution, we strive to exceed expectations and
    create magical moments that will be cherished forever.</p>

    <p>As we embark on this journey of turning dreams into reality,
    DecorX is proud to be a part of your special celebrations.
    Explore our website to discover our portfolio, browse through our services,
    and envision the possibilities for your upcoming event.
    For inquiries or to book our services, feel free to

```

➤ register.html

```

<form action="RegisterServlet" method="post">
<table>
  <tr>
    <td>
      |      FirstName:
    </td>
    <td>
      <input type="text" name="firstname" placeholder="enter your firstname">
    </td>
  </tr>
  <tr>
    <td>
      LastName:
    </td>
    <td>
      <input type="text" name="lastname" placeholder="enter your lastname">
    </td>
  </tr>
  <tr>
    <td>
      Mobile No:
    </td>
    <td>
      <input type="text" name="mobilenno" placeholder="enter your mobile no" >
    </td>
  </tr>
  <tr>
    <td>
      Email:
    </td>
    <td>

```

➤ profile.jsp

```

<%>
<%try{
    out.println("<h1>Profile</h1><br>");
    Class.forName("oracle.jdbc.driver.OracleDriver");
    conn=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","vivek597","nakka597");
    st=conn.createStatement();
    ResultSet rs=st.executeQuery("select * from userdetails where username='"+session.getAttribute("username")+"'");
    ResultSetMetaData rmd=rs.getMetaData();
    int columnCount = rmd.getColumnCount();
    while(rs.next()){
        out.println("<table >");
        for (int i = 1; i <= columnCount; i++) {
            if(i!=2)
            {
                out.println("<tr>");
                out.println("<td>"+rmd.getColumnName(i)+"</td>");
                out.println("<td>"+rs.getString(i)+"</td>");
                out.println("</tr>");
            }
        }
        out.println("</table>");
    }
    conn.close();
} catch (Exception e){
    out.println(e.getMessage());
}

```

➤ RegisterServlet.java

```
response.setContentType("text/html");
PrintWriter out=response.getWriter();
String uname=request.getParameter("uname");
String password=request.getParameter("password");
String firstname=request.getParameter("firstname");
String lastname=request.getParameter("lastname");
String email=request.getParameter("email");
String village=request.getParameter("village");
String mandal=request.getParameter("mandal");
String district=request.getParameter("district");
String address=request.getParameter("address");
String mobilenno=request.getParameter("mobilenno");
String pinno=request.getParameter("pinno");
String gender=request.getParameter("gender");
String state=request.getParameter("state");
Connection conn = null;
try {

    Class.forName("oracle.jdbc.driver.OracleDriver");
    conn=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","vivek597","nakka597");
    Statement st=conn.createStatement();
    PreparedStatement ps=conn.prepareStatement("insert into userdetails values(?,?,?,?,?,?,?,?,?,?)");
    ps.setString(1,uname);
    ps.setString(2,password);
    ps.setString(3,firstname);
    ps.setString(4,lastname);
    ps.setString(5,gender);
    ps.setString(6,mobilenno);
    ps.setString(7,email);
    ps.setString(8,address);
```

➤ UserServlet.java

```
 * @see HttpServlet#HttpServlet()
 */
public UserServlet() {
    super();
    // TODO Auto-generated constructor stub
}

/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    //response.getWriter().append("Served at: ").append(request.getContextPath());
    Statement st;
    Connection conn=null;
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();
    try {out.println("hello");
        Class.forName("oracle.jdbc.driver.OracleDriver");
        conn=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","vivek597","nakka597");
        st=conn.createStatement();
        ResultSet rs=st.executeQuery("select username,password from userdetails");
        String uname=request.getParameter("uname");
        String pass=request.getParameter("password");
        while(rs.next())
        {
```


➤ Product.java

```
package com.example;

public class Product {

    private Integer pid;
    private String pname;
    private Integer rating;
    private Double price;
    private String imgpath;
    private String description;

    public Product(Integer pid,String pname,Double price){
        this.pid=pid;
        this.pname=pname;
        this.price=price;
    }

    public Product(Integer pid,String pname,Double price,Integer rating,String imgpath,String description){
        this.pid=pid;
        this.pname=pname;
        this.rating=rating;
        this.price=price;
        this.imgpath=imgpath;
        this.description=description;
    }

    public Integer getPid()
    {
        return this.pid;
    }

    public String getPname() {
        return this.pname;
    }

    public Integer getRating() {
        return this.rating;
    }
}
```

➤ OrderServlet.java

```
String eventType=request.getParameter("eventSelect");

out.println("fname:"+fname+"lname:"+lname+"deno:"+designNo+"eventype"+eventType);
try {
    Class.forName("oracle.jdbc.driver.OracleDriver");
    Connection conn=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","1234");
    PreparedStatement ps=conn.prepareStatement("insert into orderDetails values(?,?,?,?)");
    ps.setString(1, fname);
    ps.setString(2, lname);
    ps.setInt(3,Integer.valueOf(designNo));
    ps.setString(4, eventType);
    ps.setInt(5,15999);
    boolean flag=ps.execute();
    if(!flag)
    {
        out.println("<h1>"+fname+" your order is confirmed....</h1>");
    }
    else
    {
        out.println("<h1>"+fname+" sorry your order is not confirmed....</h1>");
    }
    conn.close();
}
catch(Exception e) {
    e.printStackTrace();
    // System.out.println(e.getMessage());
}
```

➤ YourOrders.jsp

```
47      ResultSetMetaData rmd=rs.getMetaData();
48      int columnCount = rmd.getColumnCount();
49      out.println("<table >");
50      out.println("<tr>");
51      for (int i = 1; i <= columnCount; i++)
52      {
53          out.println("<th>"+rmd.getColumnName(i)+"</th>");
54      }
55      out.println("<th>Design details</th>");
56      out.println("</tr>");
57      while(rs.next()){
58          out.println("<tr>");
59          for (int i = 1; i <= columnCount; i++) {
60
61              out.println("<td>"+rs.getString(i)+"</td>");
62
63          }
64          ResultSet rsl=st.executeQuery("select p.pname,p.price,p.imagepath,p.rating from products p,orders o where o.pid=p.pid");
65          rsl.next();
66          out.println("<td>");
67          out.println("<h1>"+rsl.getString(1)+"</h1>");
68          out.println("<img src='"+rsl.getString(3)+"'>");
69          out.println("<br>Rating:"+rsl.getString(4)+"/5");
70          out.println("<br>Price:"+rsl.getString(2));
71          out.println("</td>");
72          out.println("</tr>");
73      }
74      out.println("</table>");
75
76      conn.close();
77  } catch (Exception e){
```

➤ YourCart.jsp

```
38 %>
39 <%try{
40     if(session.getAttribute("username")==null)
41         response.sendRedirect("userlogin.html");
42     out.println("<h1>YOUR CART</h1><br>");
43     Class.forName("oracle.jdbc.driver.OracleDriver");
44     conn=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","vivek597","nakka597");
45     st=conn.createStatement();
46     ResultSet rs=st.executeQuery("select username from cart where username='"+session.getAttribute("username")+"'");
47     int i=0;
48     int cnt=rs.getMetaData().getColumnCount();
49     while(i-->0){
50         out.println("<tr>");
51         rs.next();
52         ResultSet rsl=st.executeQuery("select p.pname,p.price,p.imagepath,p.rating from products p,card c where p.pid=c.pid");
53         rsl.next();
54         out.println("<td>");
55         out.println("<h1>"+rsl.getString(1)+"</h1>");
56         out.println("<img src='"+rsl.getString(3)+"'>");
57         out.println("<br>Rating:"+rsl.getString(4)+"/5");
58         out.println("<br>Price:"+rsl.getString(2));
59         out.println("</td>");
60         out.println("</tr>");
61     }
62     out.println("</table>");
63
64     conn.close();
65 } catch (Exception e){
66     out.println(e.getMessage());
67 }
68 %>
```


4.2 Interfaces

➤ User & Admin Login Interfaces



The User Login interface is a dark blue rectangular box with a yellow border. It contains the title "User Login" in white serif font. Below the title are two input fields: "username:" followed by a white box containing the placeholder text "enter your username", and "password:" followed by an empty white box. To the right of the password field is a link "Register here" in white text. At the bottom left is a button labeled "Login as User" in white text on a dark blue background.



The Admin Login interface is a dark blue rectangular box with a yellow border. It contains the title "Admin Login" in white serif font. Below the title are two input fields: "username:" followed by a white box containing the placeholder text "enter your username", and "password:" followed by an empty white box. At the bottom left is a button labeled "Login as Admin" in white text on a dark blue background.

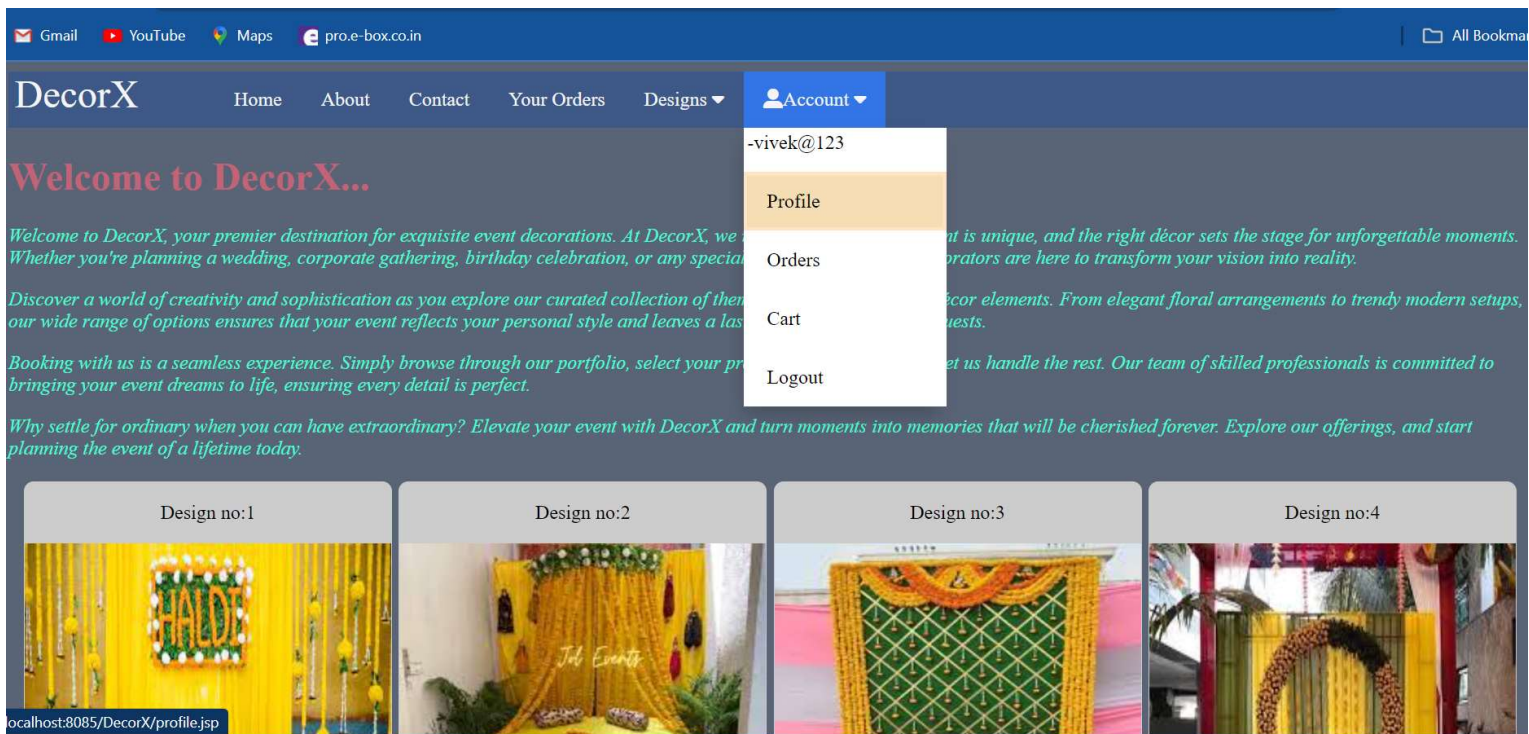
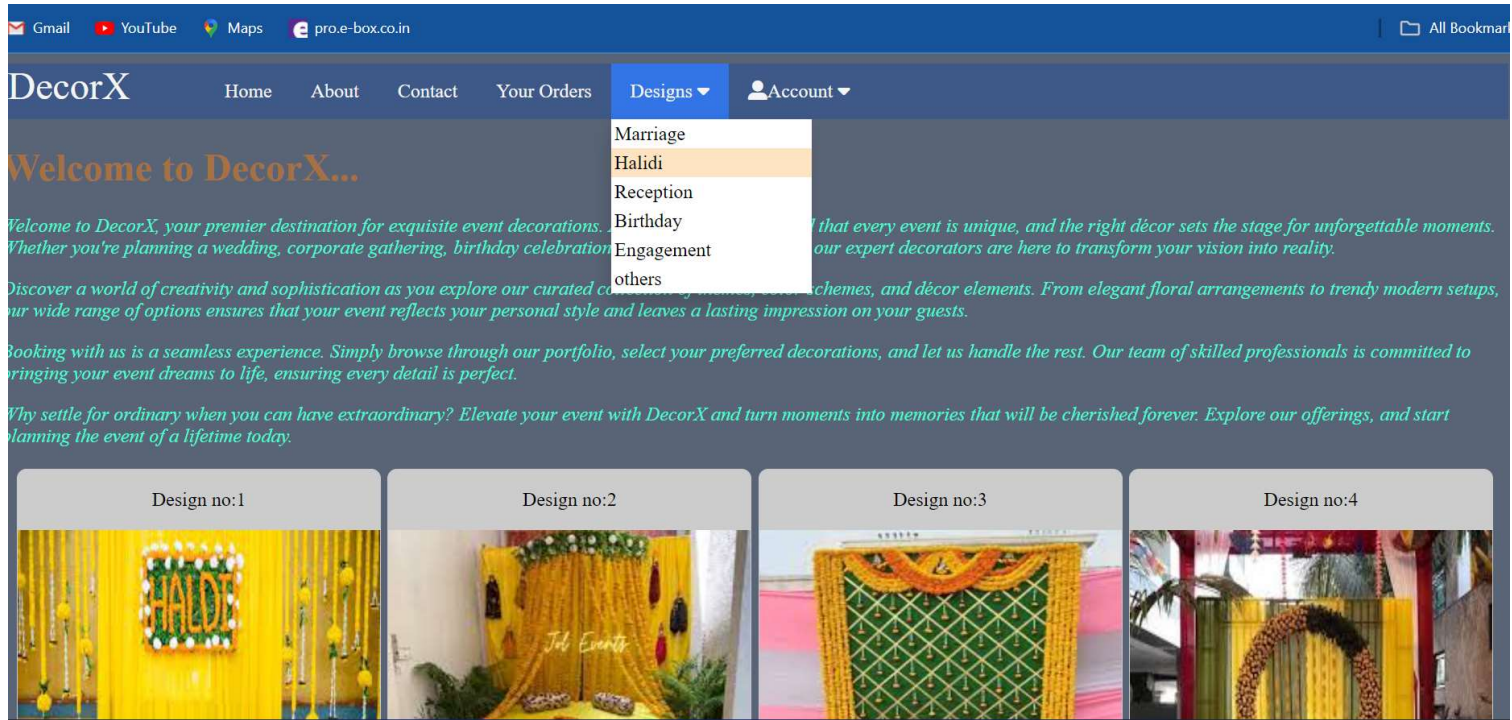
➤ Register Interface

Register

FirstName:	<input type="text" value="enter your firstname"/>
LastName:	<input type="text" value="enter your lastname"/>
Mobile No:	<input type="text" value="enter your mobile no"/>
Email:	<input type="text" value="enter your email"/>
Gender:	<input type="radio"/> male <input type="radio"/> female <input type="radio"/> others
Address:	<input type="text" value="H-no,streetno,village"/>
Village/Town:	<input type="text" value="village/town name"/>
Mandal:	<input type="text" value="mandal name"/>
District:	<input type="text" value="district name"/>
Pin:	<input type="text" value="pin number"/>
State:	<input type="text" value="state name"/>
UserName:	<input type="text" value="example@gmail.com"/>
Password:	<input type="password"/>
Confirm Password:	<input type="password"/>

[goto login here](#)

➤ Homepage with design catalog



Profile

USERNAME	vivek@123
FIRSTNAME	Vivek
LASTNAME	Nakka
GENDER	male
MOBILE	8465826966
EMAIL	viveknakka2003@gmail.com
ADDRESS	2-82,yellapgonda
VILLAGE	yellapgonda
MANDAL	palvancha
DISTRICT	kamareddy
PINNO	503112
STATE	telangana


➤ Your Orders page

YOUR ORDERS						
USERNAME	PID	BOOKEDDATE	EVENTDATE	EVENTTIME	ADDRESS	Design details
vivek@123	121	2024-02-06 00:00:00.0	2024-02-20 00:00:00.0	2024-2- 6.12.30. 0. 0	null	<div>marriage1</div> <div></div> <div>Rating:4/5 Price:8000</div>


➤ contat page

[Gmail](#) [YouTube](#) [Maps](#) [pro.e-box.co.in](#)


Contact Us




Facebook




Twitter



Insta



Whatsapp



youtube

Conclusion

In conclusion, the proposed DecorX platform combines a robust set of frontend and back-end technologies to create a user-friendly and visually appealing environment for event decoration planning. Leveraging HTML, CSS, Bootstrap, JavaScript, Java, Servlets, JSP, Oracle SQL, and JDBC, the platform aims to provide a seamless experience for users and decorators alike.

The frontend technologies ensure an engaging and responsive user interface, while the back-end technologies, including Java, Servlets, and JSP, handle the dynamic logic and server-side rendering. Oracle SQL, coupled with JDBC, forms a reliable database management system to efficiently store and retrieve user data, decorator profiles, and booking details.

References

Bootstrap Documentation:

<https://getbootstrap.com/docs/5.3/getting-started/introduction/>

Java Documentation:

<https://docs.oracle.com/en/java/javase/18/>

Servlets Documentation:

<https://docs.oracle.com/javaee%2F7%2Fapi%2F%2F/javax/servlet/Servlet.html>

JSP Documentation:

<https://docs.oracle.com/javaee/5/tutorial/doc/bnajo.html>

Oracle Database Documentation:

<https://docs.oracle.com/database/121/SQLRF/toc.htm>

JDBC Documentation:

<https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/>

In addition to them

<https://www.w3schools.com/tutorials/>

<https://www.tutorialspoint.com/index.htm>