EX NO: DATE:	RESPONSIVE WEBPAGE WITH CSS SELECTORS

AIM:

To implement a responsive webpage using CSS Selectors

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Responsive Webpage</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header class="header">
    <h1>Welcome to My Responsive Webpage</h1>
    <nav>
      <ul class="nav">
        <li><a href="#">Home</a></li>
        <li><a href="#">About</a></li>
        <li><a href="#">Services</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
    </nav>
  </header>
  <main class="main-content">
    <section class="intro">
      <h2>Introduction</h2>
      <p>This is a simple responsive webpage designed using CSS selectors.</p>
    </section>
    <section class="features">
```

```
      <h2>Features</h2>
      <ul>
        <li>Responsive Design</li>
        <li>CSS Selectors</li>
        <li>Cross-device Compatibility</li>
      </ul>
    </section>
  </main>
  <footer class="footer">
    <p>&copy; 2024 My Responsive Webpage</p>
  </footer>
</body>
</html>
```

Style.css:

```
/* Global styles */
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
  }

  /* Header styles */
  .header {
    background-color: #4CAF50;
    color: white;
    padding: 10px 20px;
    text-align: center;
  }
  .nav {
    list-style-type: none;
```

```
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
}
.nav li {
  margin: 0 10px;
}
.nav a {
  color: white;
  text-decoration: none;
  font-weight: bold;
}
.nav a:hover {
  text-decoration: underline;
}
/* Main content styles */
.main-content {
  padding: 20px;
}
.intro, .features {
  margin: 20px 0;
}
.features ul {
  list-style-type: square;
}
```

OUTPUT:

CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented by using the properties of CSS Selectors and output has been verified successfully.

EX NO: DATE:	WEBPAGE USING TRANSITION

AIM:

To implement a responsive webpage using CSS Transition

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>Colorful Transition Showcase</title>
<link rel="stylesheet" href="style.css">
</head>

<body>
<h1>Transitions in CSS</h1>
<BR></BR>
<section class="transition-buttons">
<h2>Button Transition</h2>
<button class="btn-slide">Slide Button</button>
<button class="btn-grow">Grow Button</button>
<button class="btn-fade">Fade Button</button>
</section>


<section class="transition-cards">
<h2>Card Transition</h2>

<div class="card card-rotate">
<h3>Rotating Card</h3>
</div>
```

```
<div class="card card-lift">
<h3>Lifting Card</h3>
</div>
<div class="card card-color-change">
<h3>Color Changing Card</h3>
</div>
</section>
<section class="text-transition">
<h2>Text Transitions</h2>
<p class="text-fade">Hover to see this text fade in and out.</p>
<p class="text-slide">Hover to see this text slide from the left.</p>
<p class="text-color">Hover to change the color of this text.</p>
</section>
</body>
</html>
```

Style.css:

```
* {
margin: 0;
padding: 0;
box-sizing: border-box;
}
body {
font-family: 'Arial', sans-serif; background-color: #f0f8ff; color: #333;
text-align: center; padding: 20px;
}
/* Transition Buttons */
.transition-buttons { margin-bottom: 50px;
}
.transition-buttons h2 { font-size: 30px; margin-bottom: 20px;
}
button {
```

```
padding: 15px 30px; font-size: 18px; margin: 10px; border-radius: 8px; border: none; cursor: pointer;
transition: all 0.4s ease; /* Universal transition for buttons */
}
/* Slide Button */
.btn-slide {
background-color: #f39c12;
}
/* Grow Button */
.btn-grow {
background-color: #1abc9c; color: #fff;
}
.btn-fade {
background-color: #e74c3c; color: #fff;
}
.btn-fade:hover {
opacity: 0.5; /* Fade effect on hover */
}
.transition-cards { margin-bottom: 50px;
}
.transition-cards h2 { font-size: 30px; margin-bottom: 20px;
}
/* Rotating Card */
.card-rotate:hover {
transform: rotate(10deg); /* Rotate on hover */
}
```

OUTPUT:

CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented by using the CSS transitions and output has been verified successfully

EX NO: DATE:	WEBPAGE USING ANIMATION

AIM:

To implement a responsive webpage using CSS Animation

PROGRAM:

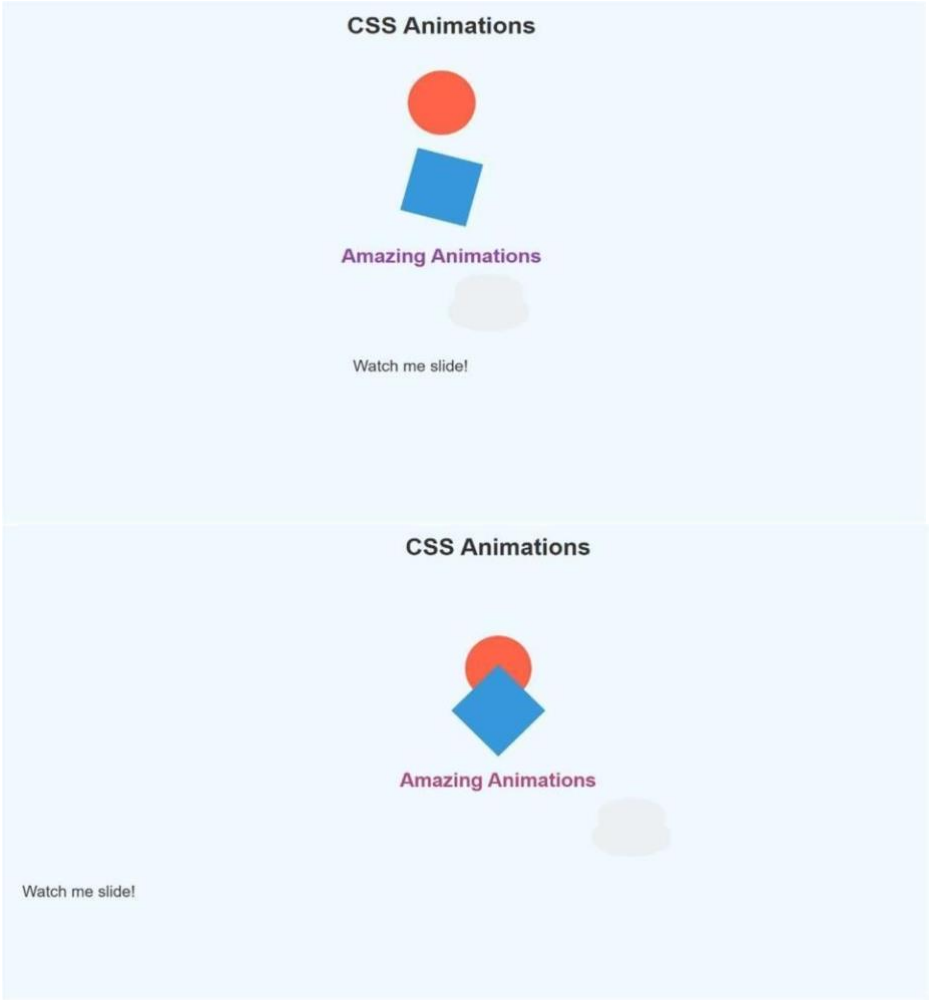
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>CSS Animations </title>
<style>
* {

margin: 0;
padding: 0;
box-sizing: border-box;
}
body {
font-family: 'Arial', sans-serif; background-color: #f0f8ff; color: #333;
text-align: center; padding: 20px;
}



h1 {
font-size: 36px;
margin-bottom: 20px;}

.ball {
width: 100px; height: 100px;
background-color: #ff6347; /* Tomato color */ border-radius: 50%;
margin: 40px auto; position: relative;
```

```
animation: bounce 2s infinite ease-in-out;
}

.square { width: 100px; height: 100px;
background-color: #3498db; /* Blue color */ margin: 40px auto;
animation: rotate 4s infinite linear;
}

@keyframes rotate { 0% {
transform: rotate(0deg);
} 100% {
transform: rotate(360deg);
}
}
.color-change { font-size: 30px; font-weight: bold;
margin: 40px auto;
animation: colorChange 3s infinite;
}
@keyframes colorChange {

0% {
color: #e74c3c;
} 50% {
color: #8e44ad;
} 100% {
color: #3498db;
}

@keyframes float { 0% {
transform: translateX(0);
} 50% {
transform: translateX(300px);
```

```
} 100% {
transform: translateX(0);
}
}
.cloud::before, .cloud::after { content: '';
position: absolute; background-color: #ecf0f1; width: 100px;
height: 50px; border-radius: 50%;
}
.cloud::before { top: -30px; left: 10px;

}
.slide-text {
font-size: 24px; margin: 40px auto;
animation: slideIn 3s infinite;
}
@keyframes slideIn { 0% {
transform: translateX(-100%);

} 50% {
transform: translateX(0);
} 100% {
transform: translateX(100%);
}
}
</style>
</head>
<body>
<h1>CSS Animations </h1>
<div class="ball"></div>
<div class="square"></div>
<p class="color-change">Amazing Animations</p>
```

```
<div class="cloud"></div>
<p class="slide-text">Watch me slide!</p>
</body>
</html>
```

OUTPUT:



CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented by using the CSS animations and output has been verified successfully.

EX NO: DATE:	WEBPAGE USING BASIC CONCEPTS OF JAVASCRIPT

AIM:

To implement a webpage that uses the basic concepts of Javascript

PROGRAM:

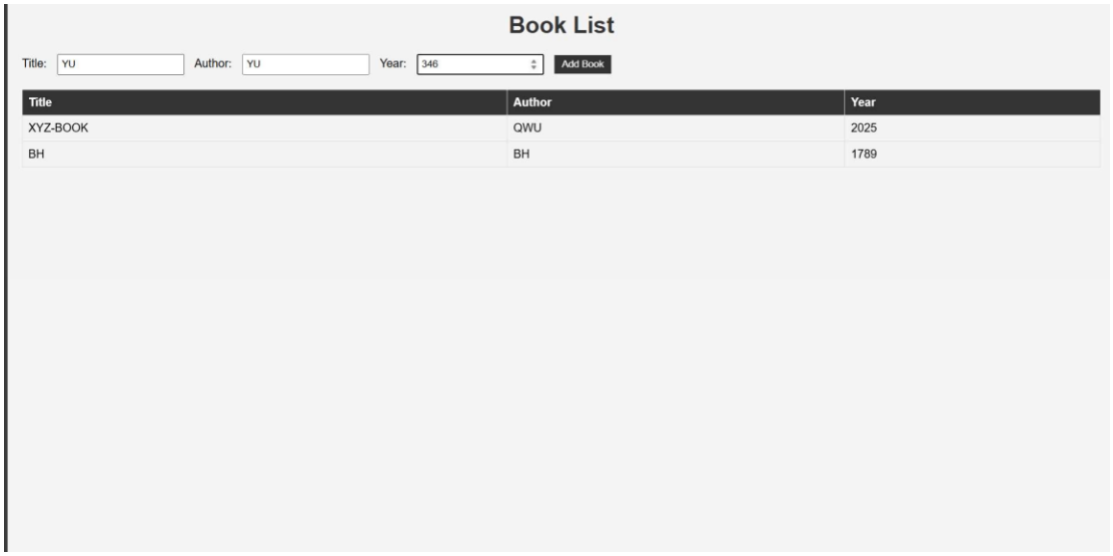
Index.html:

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

font-family: Arial, sans-serif; margin: 20px;
background-color: #f4f4f4;
}
h1 {
text-align: center; color: #333;
}
#book-list {
margin-top: 20px
table {
width: 100%;
border-collapse: collapse;

}
#book-form {
margin-bottom: 20px;
}
label {
```

```
margin-right: 10px;
}
button:hover { background-color: #555;
}
</style>
</head>
<body>
<h1>Book List</h1>
<div id="book-form">
<label for="title">Title:</label>
<input type="text" id="title" placeholder="Book Title" required>
<label for="author">Author:</label>
<input type="text" id="author" placeholder="Book Author" required>
<label for="year">Year:</label>
<input type="number" id="year" placeholder="Publication Year" required>
<button onclick="addBook()">Add Book</button>
</div>
<div id="book-list">
<table>
<thead>
<tr>
<th>Title</th>
<th>Author</th>
<th>Year</th>
</tr>
</thead>
<tbody id="books-table">
<!-- Books will be dynamically added here -->
</tbody>
</table>
</div>
<script>
```

```
// Array to store the list of books const books = [];

function Book(title, author, year) { this.title = title;

this.author = author; this.year = year;

}

function displayBooks() {

const booksTable = document.getElementById('books-table'); booksTable.innerHTML = '';

books.forEach(book => {

const row = document.createElement('tr'); const titleCell = document.createElement('td');
titleCell.textContent = book.title; booksTable.appendChild(row);

});

}

function addBook() {

const title = document.getElementById('title').value; const year =
document.getElementById('year').value; if (title === '' || author === '' || year === '') {

alert('Please fill in all fields.'); return;

}

document.getElementById('author').value = ''; document.getElementById('year').value = '';
displayBooks();

}

</script>

</body>

</html>
```

OUTPUT:



CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented by using the basic concepts of javascript and the output has been verified successfully.

EX NO: DATE:	WEBPAGE WITH JAVASCRIPT ASYNCHRONOUS FUNCTIONS

AIM:

To implement a webpage using the Javascript Asynchronous functions

PROGRAM:

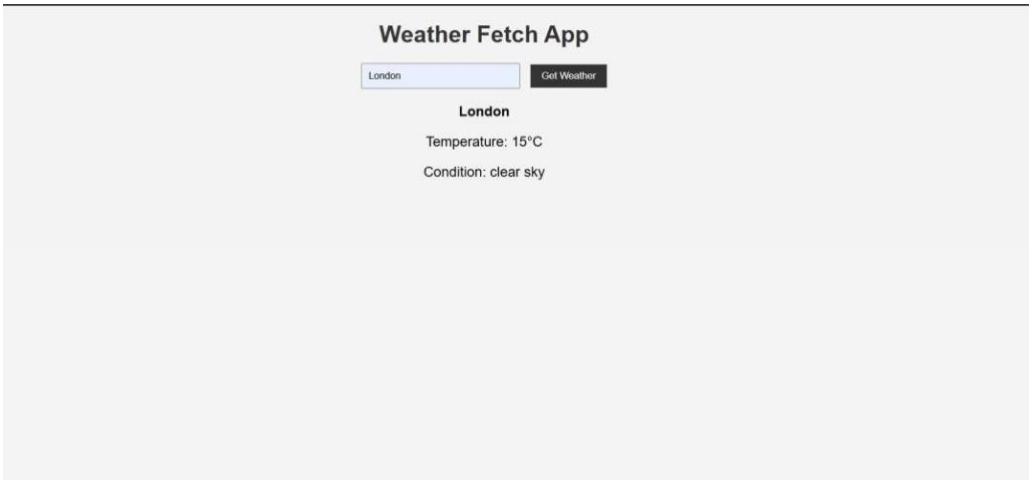
Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Weather Fetch App </title>
<style>
body {
font-family: Arial, sans-serif; background-color: #f4f4f4; margin: 20px;
}
h1 {
text-align: center; color: #333;
}
#weather-form { text-align: center;
margin-bottom: 20px;
}
input[type="text"] { padding: 8px; width: 200px;
margin-right: 10px;
button {
padding: 8px 15px; background-color: #333; color: white;
border: none; cursor: pointer;
}
.loading { color: #999;
}
.error {
```

```
color: red;
}
</style>
</head>
<body>
<h1>Weather Fetch App</h1>
<div id="weather-form">
<input type="text" id="city" placeholder="Enter city name" required>
<button onclick="getWeather()">Get Weather</button>
</div>
<div id="weather-result"></div>
<script>
const mockWeatherData = {
"London": { temp: 15, description: "clear sky" }, "New York": { temp: 10, description: "cloudy" },
"Paris": { temp: 18, description: "sunny" }, "Tokyo": { temp: 22, description: "light rain" }
};
function mockFetchWeather(city) {
return new Promise((resolve, reject) => { setTimeout(() => {
// Convert the input city to lowercase to avoid case mismatch issues const cityFormatted =
city.trim().toLowerCase();
const cityData = Object.keys(mockWeatherData).find( key => key.toLowerCase() === cityFormatted
);
if (cityData) { resolve(mockWeatherData[cityData]);
} else {
reject("City not found");
}
}, 1000); // Simulate 1 second delay
});
}
async function getWeather() {
const city = document.getElementById('city').value;
const weatherResult = document.getElementById('weather-result'); weatherResult.innerHTML = '';
if (city.trim() === '') {
```

```
weatherResult.innerHTML = '<p class="error">Please enter a city name.</p>'; return;
}
weatherResult.innerHTML = '<p class="loading">Loading...</p>'; try {
const weather = await mockFetchWeather(city); weatherResult.innerHTML = `
<p><strong>${city}</strong></p>
<p>Temperature: ${weather.temp}°C</p>
<p>Condition: ${weather.description}</p>;
} catch (error) {
weatherResult.innerHTML = `<p class="error">${error}</p>`;
}
}
</script>
</body>
</html>
```

OUTPUT:



CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented using javascript asynchronous functions and the output has been verified successfully.

EX NO: DATE:	WEBPAGE USING REACT

AIM:

To implement a webpage using the React JS

PROGRAM:

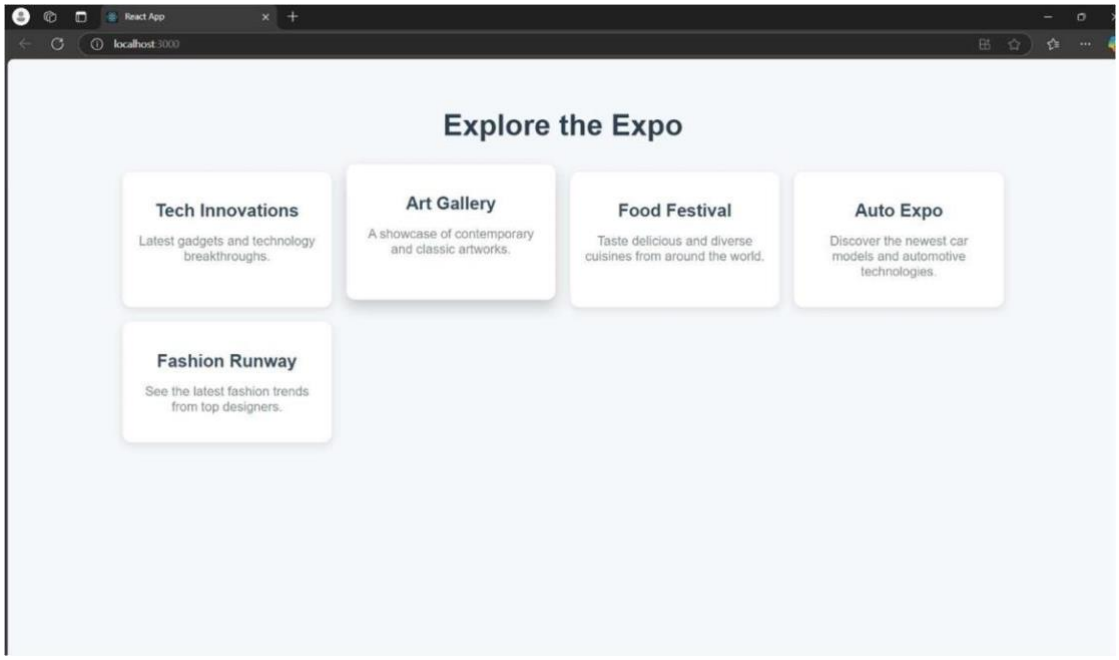
App.js:

```
import React from 'react'; function App() {

const expoItems = [

{ id: 1, title: "Tech Innovations", description: "Latest gadgets and technology breakthroughs." },

{ id: 2, title: "Art Gallery", description: "A showcase of contemporary and classic artworks." },

{ id: 3, title: "Food Festival", description: "Taste delicious and diverse cuisines from around the world." },

{ id: 4, title: "Auto Expo", description: "Discover the newest car models and automotive technologies." },

{ id: 5, title: "Fashion Runway", description: "See the latest fashion trends from top designers." },

];

const containerStyle = { padding: "40px", backgroundColor: "#f5f7fa", fontFamily: "Arial, sans-serif", minHeight: "100vh", textAlign: "center"

};


const headerStyle = { color: "#2c3e50", marginBottom: "40px", fontSize: "2.5em", fontWeight: "bold",


};

const gridStyle = { display: "grid",

gridTemplateColumns: "repeat(auto-fit, minmax(250px, 1fr))", gap: "20px",

maxWidth: "1200px", margin: "0 auto",

};

const cardStyle = { backgroundColor: "#ffffff", padding: "20px", borderRadius: "12px",

boxShadow: "0px 4px 12px rgba(0, 0, 0, 0.1)", transition: "transform 0.3s ease, box-shadow 0.3s ease", cursor: "pointer"
```

```
};

const cardHoverStyle = { transform: "translateY(-10px)",
boxShadow: "0px 10px 20px rgba(0, 0, 0, 0.2)"
};
const titleStyle = { fontSize: "1.5em", color: "#34495e", marginBottom: "10px", fontWeight: "bold"
};
const descriptionStyle = { color: "#7f8c8d", fontSize: "1.1em"
};
const handleMouseEnter = (e) => {

Object.assign(e.currentTarget.style, cardHoverStyle);
};
const handleMouseLeave = (e) => { Object.assign(e.currentTarget.style, cardStyle);
};
return (
<div style={containerStyle}>
<h1 style={headerStyle}>Explore the Expo</h1>
<div style={gridStyle}>

{expoItems.map(item => (
<div key={item.id} style={cardStyle}
onMouseEnter={handleMouseEnter} onMouseLeave={handleMouseLeave>
<h2 style={titleStyle}>{item.title}</h2>
<p style={descriptionStyle}>{item.description}</p>
</div>
))}
</div>
</div>
);

}
```

export default App;

OUTPUT:



CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented using React JS and the output is verified successfully

EX NO: DATE:	WEBPAGE USING REACT, NODE AND EXPRESS

AIM:

To implement a webpage using the React JS, Node and Express.

PROGRAM:

Server.js:

```
const express = require('express'); const cors = require('cors'); const app = express(); app.use(cors());
const books = [
{
id: 1,
title: 'The Great Gatsby', author: 'F. Scott Fitzgerald', price: 12.99,
genre: 'Classic',
image: 'https://example.com/gatsby.jpg'
},
{
id: 2,
title: 'To Kill a Mockingbird', author:  'Harper Lee', price: 14.50,
genre: 'Fiction',
image: 'https://example.com/mockingbird.jpg'
},
{
id: 3,
title: 'Clean Code', author: 'Robert C. Martin', price: 39.99,
genre 'Programming',
image: 'https://example.com/cleancode.jpg'
}
]
app.get('/api/books', (req, res) => { res.json(books);
});
const PORT = process.env.PORT || 5000; app.listen(PORT, () => {
```

```
console.log(`Server running on port ${PORT}`);
});
```

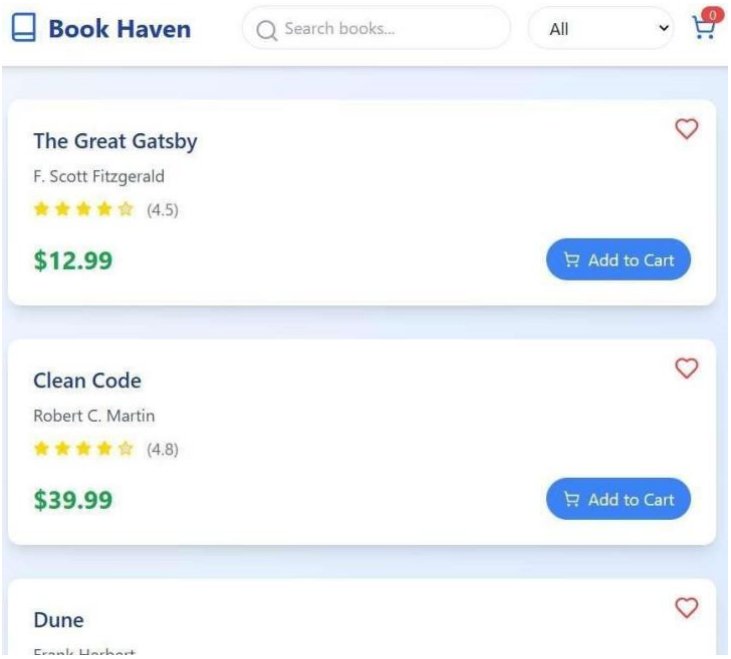
Bookstore.js:

```
import React, { useState, useEffect } from 'react';
import { ShoppingCart, Book, Heart, Star, Filter, Search } from 'lucide-react'; function BookStore() {
const [books, setBooks] = useState([]); const [cart, setCart] = useState([]);
const [searchTerm, setSearchTerm] = useState('');
const [selectedGenre, setSelectedGenre] = useState('All'); const [favorites, setFavorites] =
useState({});
const genres = [
'All', 'Fiction', 'Non-Fiction', 'Science', 'Technology', 'History', 'Romance', 'Mystery', 'Fantasy'
]
useEffect(() => { const mockBooks = [
{
id:1,
title: 'The Great Gatsby', author: 'F. Scott Fitzgerald', genre: 'Fiction',
price: 1.99,
rating: 4.5
},
{
id: 2,
title: 'Clean Code', author: 'Robert C. Martin', genre:   'Technology', price: 39.99,
rating: 4.8
},
{
id: 3,
title: 'Dune',
author: 'Frank Herbert', genre: 'Science Fiction', price: 15.99,
rating: 4.7
},
// Add more books...
];
```

```
setBoks(mockBooks);
}, []);
<nav className="bg-white shadow-md p-4 flex justify-between items-center">
<div className="flex items-center">
<Book className="mr-2 text-blue-600" size={30} />
<h1 className="text-2xl font-bold text-blue-800">Book Haven</h1>
</div>
<div className="flex items-center space-x-4">
<div className="relative">
<input type="text"
placeholder="Search books..." value={searchTerm}
onChange={(e) => setSearchTerm(e.target.value)} className="pl-10 pr-4 py-2 border rounded-full
w-64"
{genres.map(genre => (
<option key={genre} value={genre}>{genre}</option>
))}
</select>
<div className="relative">
<ShoppingCart className="text-blue-600" size={24} />
<span className="absolute -top-2 -right-2 bg-red-500 text-white text-xs rounded-full px-
2">
{cart.reduce((total, item) => total + (item.quantity || 1), 0)}
</span>
</div>
</div>
</nav>
<div className="container mx-auto px-4 py-8">
<div className="grid md:grid-cols-4 gap-8">
{filteredBooks.map(book => (
<div key={book.id}
className="bg-white rounded-xl shadow-lg p-6 transform transition hover:scale-105 relative">
<div className="flex items-center mb-2">
{[1,2,3,4,5].map(star => (
```

```
<Star key={star} size={16}
fill={star <= book.rating ? "#FFD700" : "#E0E0E0"} stroke="#FFD700"
className="mr-1"
/>
))
))}
</div>
</div>
<footer className="bg-blue-900 text-white py-8 mt-8">
<div className="container mx-auto text-center">
<p>&copy; 2024 Book Haven. All rights reserved.</p>
<div className="mt-4 space-x-4">
<a href="#" className="hover:text-blue-300">About</a>
<a href="#" className="hover:text-blue-300">Contact</a>
<a href="#" className="hover:text-blue-300">Privacy Policy</a>

</div>
</div>
</footer>
</div>
);
}

export default BookStore;
```

OUTPUT:



CLASS PERFORMANCE	
VIVA	
RECORD	
TOTAL	

RESULT:

Thus, the above website has been implemented using React,Node.js and Express.js and the output is verified successfully.