Name: Sopariwala Shreya H.

Roll No: 68

Course: M.Sc. (ICT) Sem 1

Subject: 101 Frontend Development using React js.

Practical Assignment: 2

1. Create a simple HTML form with fields for name, email, and a submit button. Field name and email should have relevant validation. Use jQuery to prevent the form from submitting when the submit button is clicked and instead display an alert with the form data.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Styled Form with Validation</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

background-color: #ebb3b8;

display: grid;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.form-container {

background-color: white;

padding: 50px;

border-radius: 20px;

box-shadow: 0 8px 10px rgba(211, 174, 174, 0.1);

width: 400px;

height: 300px;

}

.form-container h2 {

margin-bottom: 5px;

text-align: center;

color: #db9090;

}

.form-container label {

display: block;

margin-bottom: 10px;

color: #be8f8f;

}

.form-container input {

width: 100%;

padding: 10px;

margin-bottom: 15px;

border: 1px solid #bc8c8c;

border-radius: 5px;

box-sizing: border-box;

}

.form-container button {

width: 100%;

padding: 10px;

background-color: #d29696;

color: rgb(9, 0, 0);

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px;

}

.form-container button:hover {

background-color: #b4717b;

}

</style>

<script>

$(document).ready(function () {

$("#myForm").submit(function (event) {

event.preventDefault(); // Prevent form submission

var name = $("#name").val();

var email = $("#email").val();

if (name === "") {

alert("Name is required.");

return;

}

if (email === "") {

alert("Email is required.");

return;

}

if (!validateEmail(email)) {

alert("Please enter a valid email address.");

return;

}

alert("Name: " + name + "\nEmail: " + email);

});

function validateEmail(email) {

var re = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

return re.test(email);

}

});

</script>

</head>

<body>

<div class="form-container">

<h2>Contact Form</h2>

<form id="myForm">

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

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

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

<input type="email" id="email" name="email" required>

<button type="submit">Submit</button>

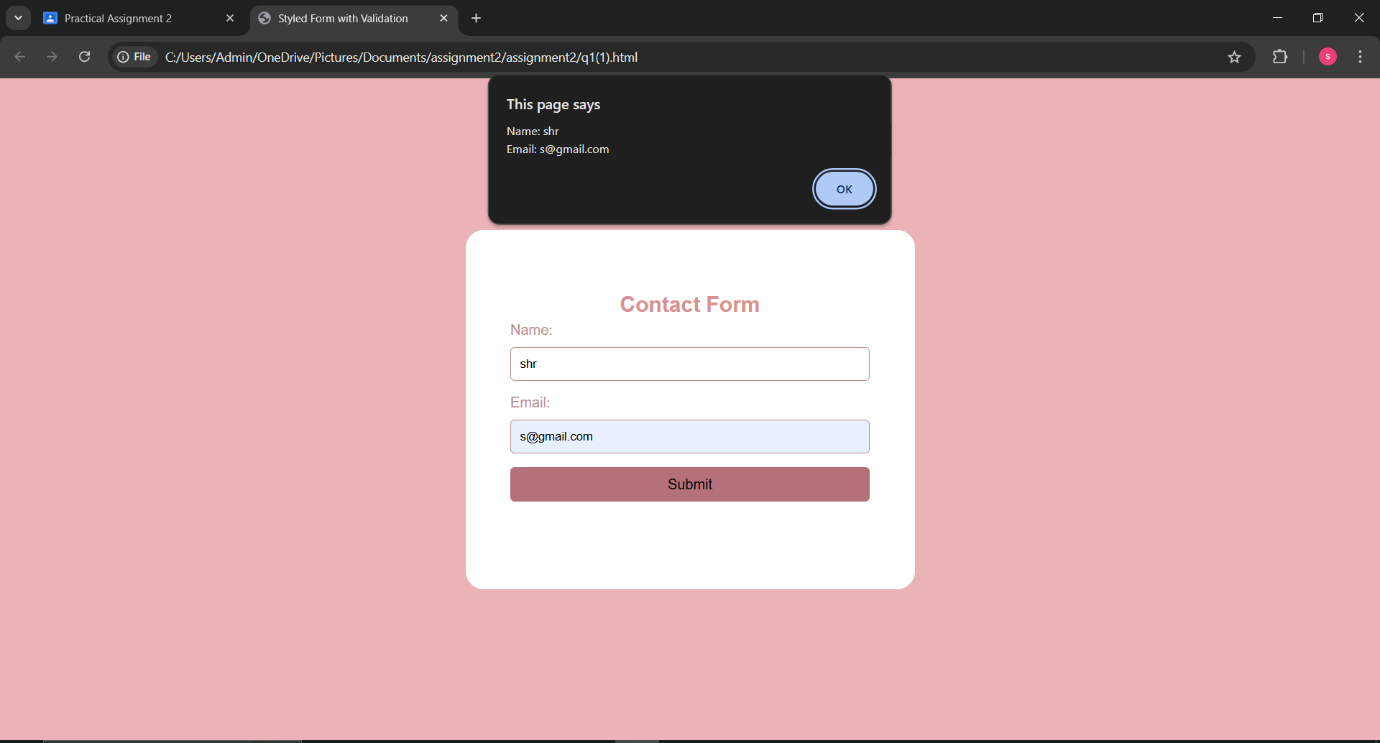
</form>

</div>

</body>

</html>

Output:



2. Create an HTML page that includes two dropdown menus (combos): one for selecting states and another for displaying cities. When the user selects a state from the primary combo, the secondary combo (cities) should dynamically update to display cities relevant to the selected state.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>State and City Dropdowns</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

background-color: #a2a2ef;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.form-container {

background-color: white;

padding: 20px;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(113, 39, 39, 0.1);

width: 350px;

}

.form-container h2 {

text-align: center;

color: #333;

}

.form-container label {

display: block;

margin-bottom: 5px;

color: #555;

}

.form-container select {

width: 100%;

padding: 10px;

margin-bottom: 15px;

border: 1px solid #ccc;

border-radius: 5px;

box-sizing: border-box;

}

.form-container button {

width: 100%;

padding: 10px;

background-color: #0bfa37;

color: rgb(13, 0, 0);

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px;

}

.form-container button:hover {

background-color: #85e36b;

}

</style>

<script>

$(document).ready(function () {

var stateCityMap = {

"Maharashtra": ["Mumbai", "Pune", "Nagpur"],

"Karnataka": ["Bangalore", "Mysore", "Mangalore"],

"Tamil Nadu": ["Chennai", "Coimbatore", "Madurai"],

"Uttar Pradesh": ["Lucknow", "Kanpur", "Varanasi"],

"Rajasthan": ["Jaipur", "Udaipur", "Jodhpur"],

"Gujarat": ["Ahmedabad", "Surat", "Vadodara"],

"Kerala": ["Thiruvananthapuram", "Kochi", "Kozhikode"],

};

$("#state").change(function () {

var selectedState = $(this).val();

var citiesDropdown = $("#city");

citiesDropdown.empty();

if (stateCityMap[selectedState]) {

$.each(stateCityMap[selectedState], function (index, city) {

citiesDropdown.append(new Option(city, city));

});

} else {

citiesDropdown.append(new Option("Select a state first", ""));

}

});

});

</script>

</head>

<body>

<div class="form-container">

<h2>Select State and City</h2>

<form id="locationForm">

<label for="state">State:</label>

<select id="state" name="state" required>

<option value="">Select a state</option>

<option value="Maharashtra">Maharashtra</option>

<option value="Karnataka">Karnataka</option>

<option value="Tamil Nadu">Tamil Nadu</option>

<option value="Uttar Pradesh">Uttar Pradesh</option>

<option value="Rajasthan">Rajasthan</option>

<option value="Gujarat">Gujarat</option>

<option value="Kerala">Kerala</option>

</select>

<label for="city">City:</label>

<select id="city" name="city" required>

<option value="">Select a state first</option>

</select>

<button type="submit">Submit</button>

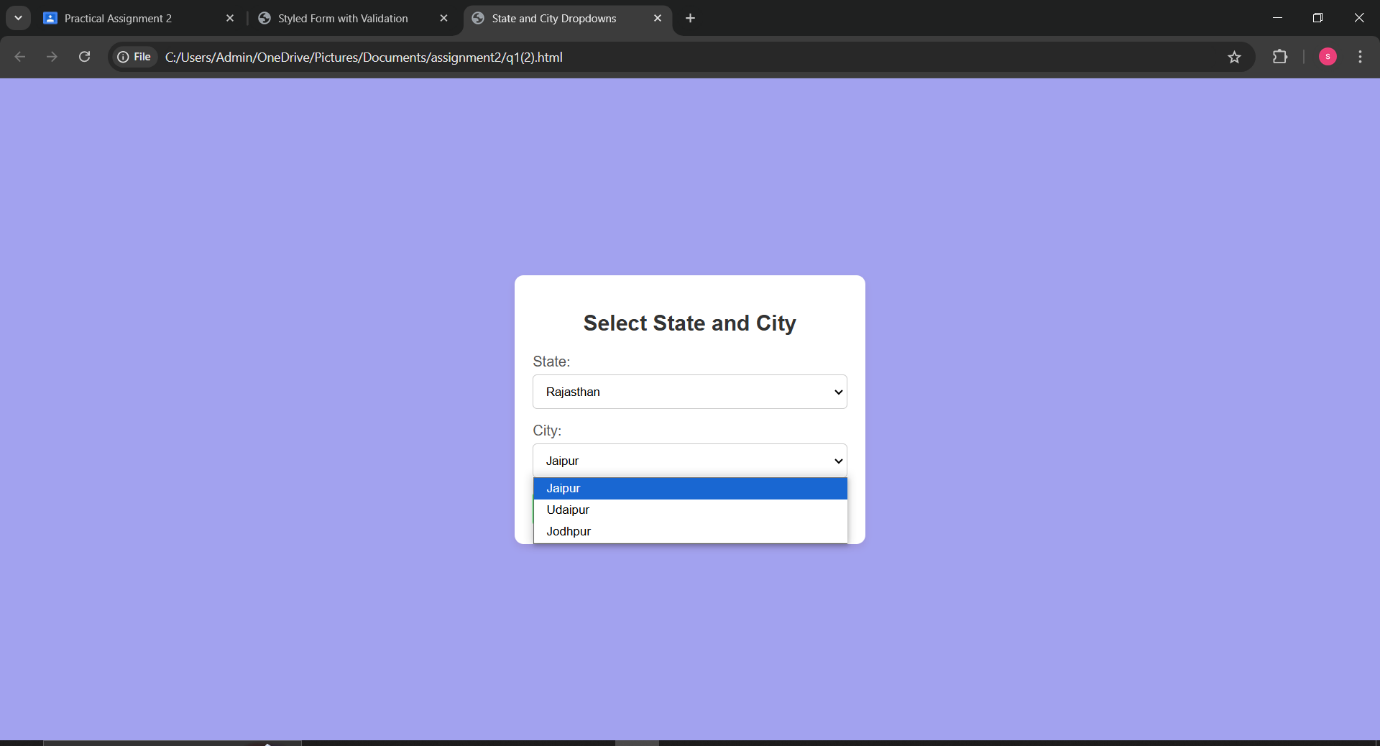
</form>

</div>

</body>

</html>

Output:



3. Write jQuery code to toggle the visibility of a div element when a button is clicked. The div should start as hidden, and clicking the button should alternate between showing and hiding it.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Toggle Div Visibility</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f9;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.container {

text-align: center;

padding: 20px;

display: block;

height: 200px;

width: 400px;

background-color: yellow;

}

.toggle-div {

display: none;

padding: 20px;

margin-top: 40px;

border-radius: 5px;

background-color: #007bff;

color: white;

font-size: 18px;

text-align: center;

}

button {

padding: 10px 20px;

background-color: #28a745;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px;

}

button:hover {

background-color: #218838;

}

</style>

<script>

$(document).ready(function () {

// Toggle the visibility of the div when the button is clicked

$("#toggleButton").click(function () {

$(".toggle-div").toggle(); // Toggles visibility (show/hide)

});

});

</script>

</head>

<body>

<div class="container">

<button id="toggleButton">Click to Toggle Div</button>

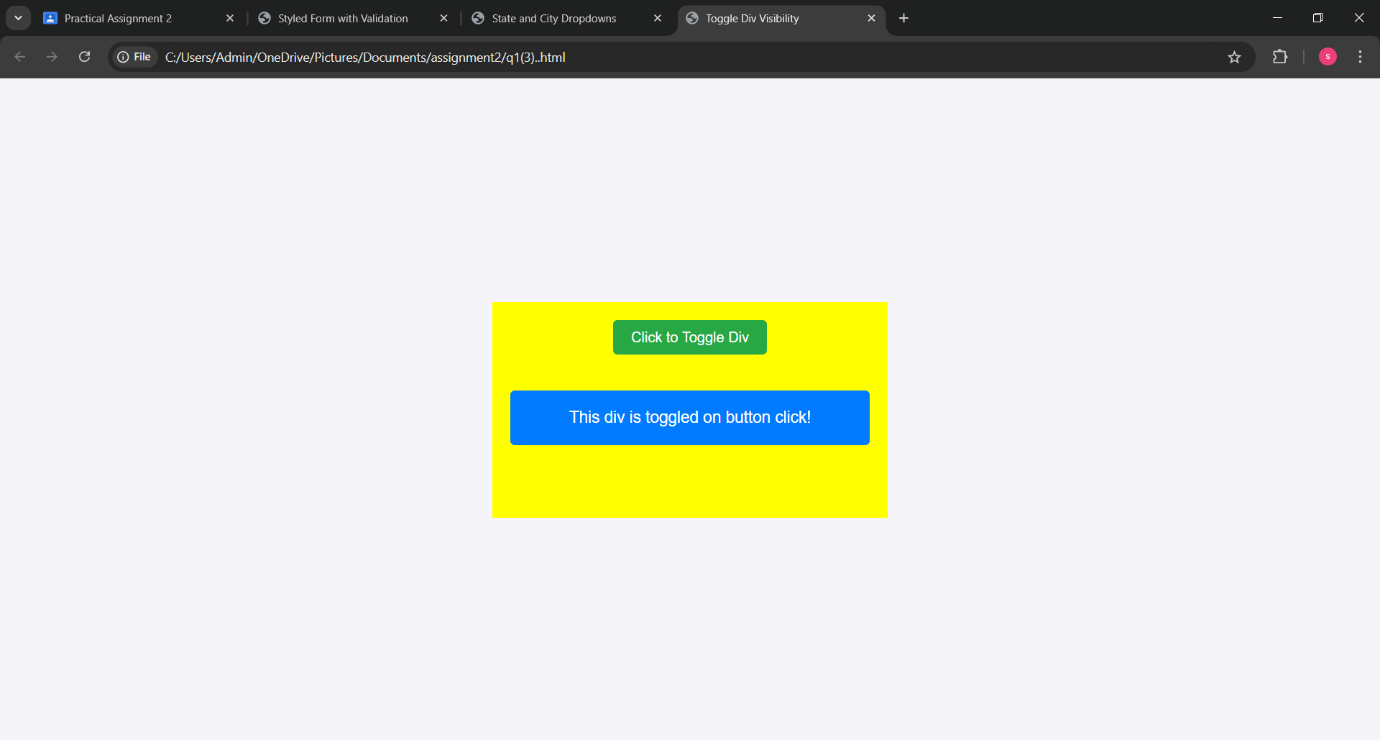
<div class="toggle-div">This div is toggled on button click! </div>

</div>

</body>

</html>

Output:



4. Create a basic image slider using jQuery. Users should be able to click “Next” and “Previous” buttons to navigate between a series of images displayed on a webpage.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Basic Image Slider</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f9;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.slider-container {

width: 500px;

position: relative;

}

.slider-container img {

width: 100%;

height: auto;

display: none;

}

.slider-container img.active {

display: block;

}

.buttons {

position: relative;

top: 50%;

width: 100%;

display: flex;

justify-content: space-between;

transform: translateY(-50%);

}

.buttons button {

background-color: #007bff;

border: none;

color: white;

padding: 10px 20px;

cursor: pointer;

font-size: 16px;

border-radius: 5px;

opacity: 0.7;

}

.buttons button:hover {

opacity: 1;

}

</style>

<script>

$(document).ready(function () {

var currentImageIndex = 0;

var images = $(".slider-container img");

var totalImages = images.length;

images.eq(currentImageIndex).addClass("active");

$("#next").click(function () {

images.eq(currentImageIndex).removeClass("active");

currentImageIndex = (currentImageIndex + 1) % totalImages;

images.eq(currentImageIndex).addClass("active");

});

$("#prev").click(function () {

images.eq(currentImageIndex).removeClass("active");

currentImageIndex = (currentImageIndex - 1 + totalImages) % totalImages;

images.eq(currentImageIndex).addClass("active");

});

});

</script>

</head>

<body>

<div class="slider-container">

<!-- Image slides -->

<img src="img/pi.jpg" alt="Image 1" class="active">

<img src="img/Pink\_flower.jpg" alt="Image 2">

<img src="img/pt.jpeg" alt="Image 3">

<img src="img/pu.jpg" alt="Image 4">

<div class="buttons">

<button id="prev">Previous</button>

<button id="next">Next</button>

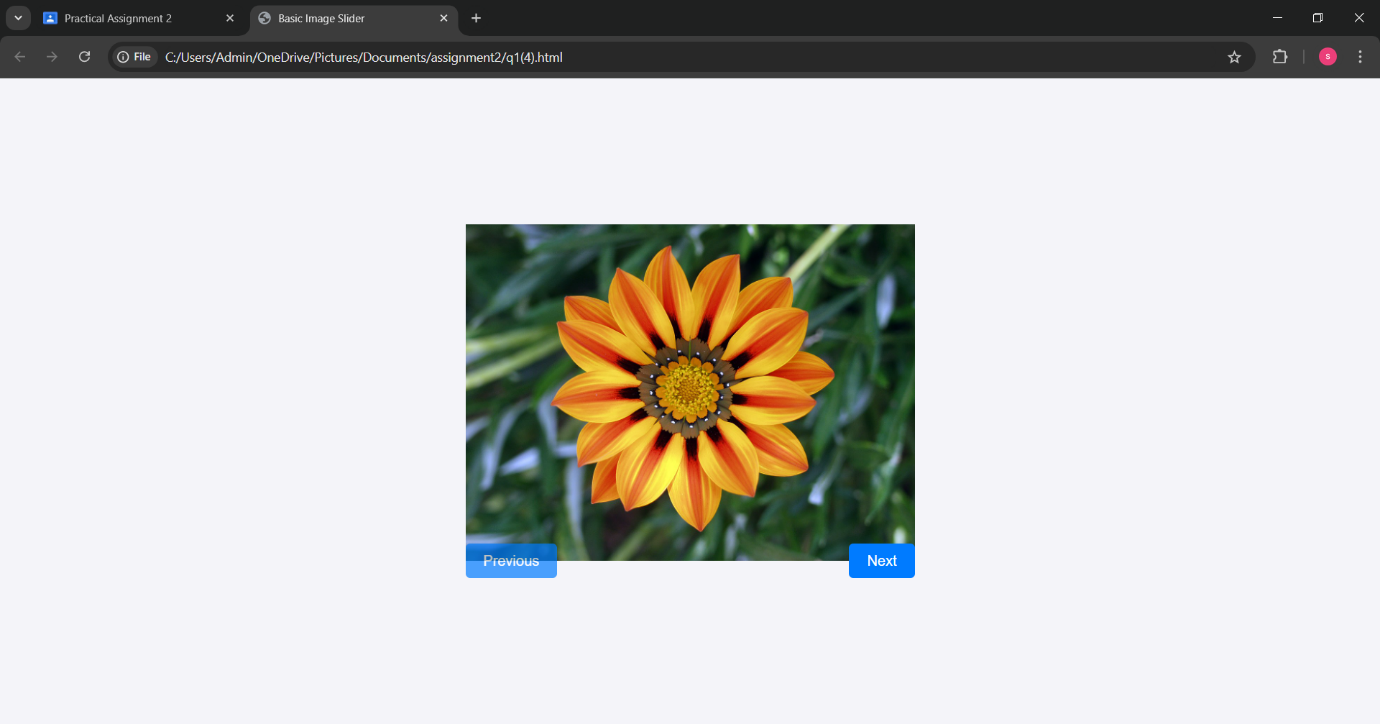
</div>

</div>

</body>

</html>

Output:



5. Implement a simple client-side validation using jQuery. Check if a text input field is empty

when a form is submitted. If it’s empty, display an error message next to the input field.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Client-Side Validation</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f9;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.form-container {

background-color: white;

padding: 20px;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

width: 350px;

}

.form-container h2 {

text-align: center;

color: #333;

}

.form-container label {

display: block;

margin-bottom: 5px;

color: #555;

}

.form-container input[type="text"] {

width: 100%;

padding: 10px;

margin-bottom: 10px;

border: 1px solid #ccc;

border-radius: 5px;

box-sizing: border-box;

}

.error-message {

color: red;

font-size: 14px;

display: none;

margin-bottom: 10px;

}

.form-container button {

width: 100%;

padding: 10px;

background-color: #007bff;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px;

}

.form-container button:hover {

background-color: #0056b3;

}

</style>

<script>

$(document).ready(function () {

$("#myForm").submit(function (event) {

// Clear previous error message

$(".error-message").hide();

var userName = $("#name").val();

if (userName.trim() === "") {

$("#nameError").show();

event.preventDefault();

}

});

});

</script>

</head>

<body>

<div class="form-container">

<h2>Form with Validation</h2>

<form id="myForm">

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

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

<span id="nameError" class="error-message">This field is required.</span>

<button type="submit">Submit</button>

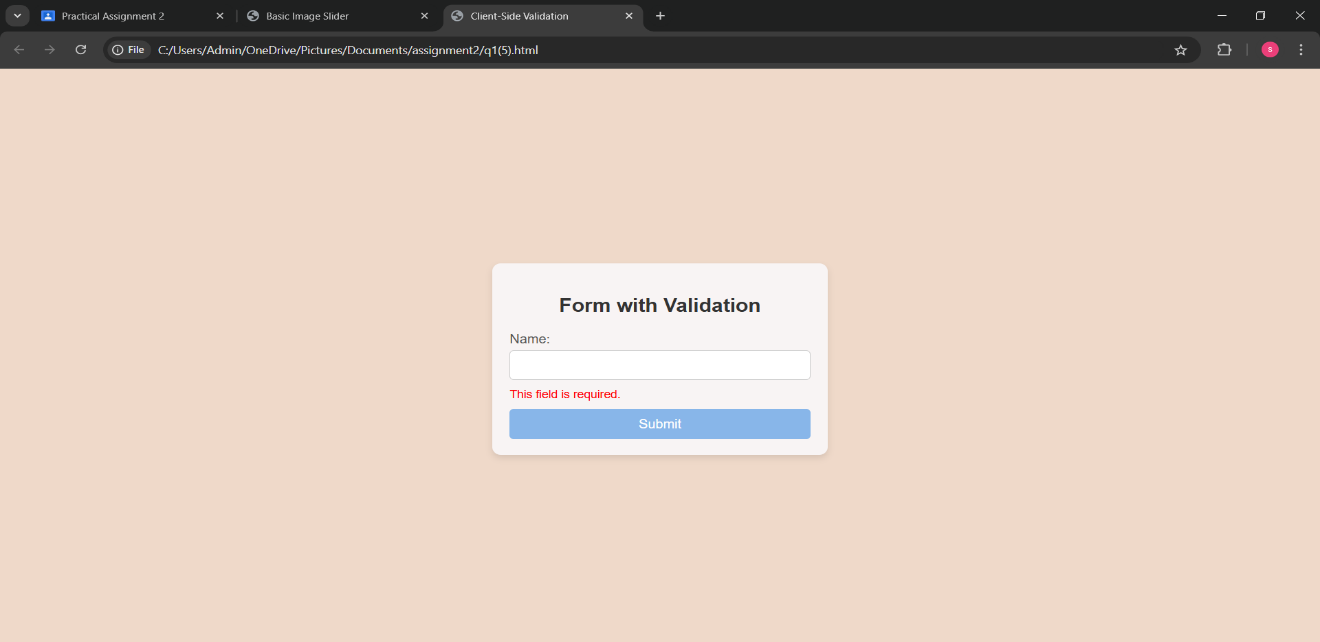
</form>

</div>

</body>

</html>

Output:



6. Develop a webpage that retrieves and displays data from a public JSON API (e.g., a list of

books, movies, or weather information) using AJAX. Display this data in a visually appealing

manner on the page.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Book List</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #e29b9b;

margin: 0;

padding: 0;

}

.container {

max-width: 900px;

margin: 50px auto;

background-color: rgb(235, 220, 220);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

h1 {

text-align: center;

color: #333;

margin-bottom: 20px;

}

.book-list {

display: flex;

flex-wrap: wrap;

gap: 20px;

justify-content: center;

}

.book {

background-color: #fff;

padding: 15px;

width: 200px;

border: 1px solid #ddd;

border-radius: 5px;

text-align: center;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

transition: transform 0.3s;

}

.book:hover {

transform: translateY(-5px);

}

.book-title {

font-size: 18px;

font-weight: bold;

margin-bottom: 10px;

}

.book-author {

color: #555;

font-size: 14px;

margin-bottom: 5px;

}

.book-year {

color: #777;

font-size: 12px;

}

</style>

</head>

<body>

<div class="container">

<h1>Book List</h1>

<div id="book-list" class="book-list">

<!-- Book data will be injected here -->

</div>

</div>

<script src="q1.js"></script>

</body>

</html>

document.addEventListener("DOMContentLoaded", function() {

const bookList = document.getElementById("book-list");

const books = [

{ title: "The Great Gatsby", author: "F. Scott Fitzgerald", year: 1925 },

{ title: "1984", author: "George Orwell", year: 1949 },

{ title: "To Kill a Mockingbird", author: "Harper Lee", year: 1960 },

{ title: "The Catcher in the Rye", author: "J.D. Salinger", year: 1951 }

];

setTimeout(() => {

displayBooks(books);

}, 1000); // 1 second delay to simulate API response

function displayBooks(books) {

let output = '';

books.forEach(book => {

output += `

<div class="book">

<div class="book-title">${book.title}</div>

<div class="book-author">by ${book.author}</div>

<div class="book-year">Published: ${book.year}</div>

</div>

`;

});

bookList.innerHTML = output;

}

});

Json File:

{"book-list":[

{

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"year": 1925

},

{

"title": "1984",

"author": "George Orwell",

"year": 1949

},

{

"title": "To Kill a Mockingbird",

"author": "Harper Lee",

"year": 1960

},

{

"title": "The Catcher in the Rye",

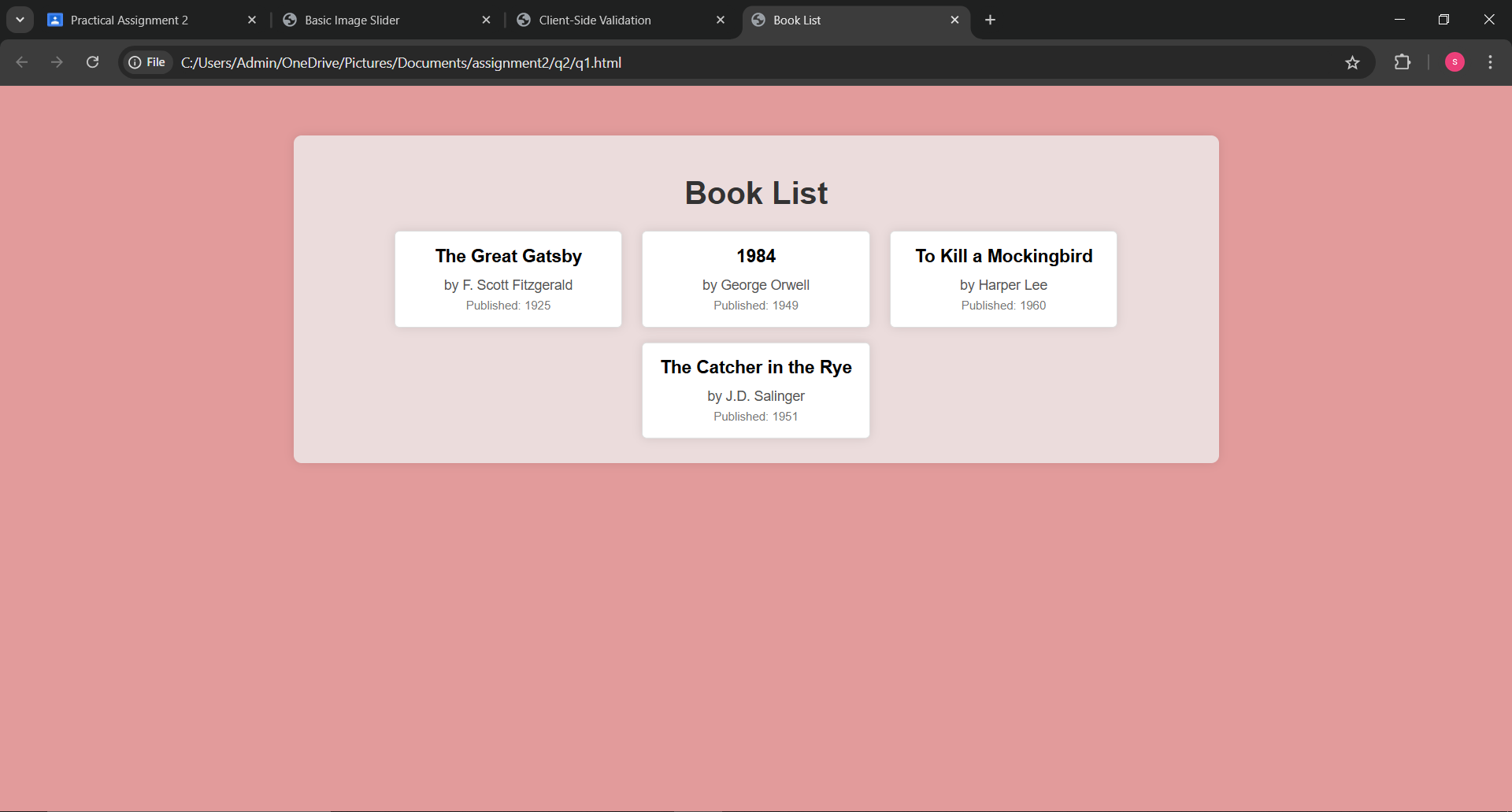
"author": "J.D. Salinger",

"year": 1951

}

]}

Output:



7. Create an autocomplete search feature using AJAX and jQuery. When users start typing a

query into a search input field, dynamically fetch and display matching results from an API

as dropdown suggestions.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Autocomplete Search</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

background-color: #bacbd3;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

.search-container {

position: relative;

width: 400px;

}

#search-input {

width: 100%;

padding: 10px;

border: 1px solid #a8d3f3;

border-radius: 5px;

font-size: 16px;

}

.suggestions-list {

position: absolute;

top: 100%;

left: 0;

right: 0;

background-color: #f7f8f9;

border: 1px solid #ccc;

border-top: none;

border-radius: 0 0 5px 5px;

max-height: 200px;

overflow-y: auto;

z-index: 999;

display: none;

}

.suggestions-list div {

padding: 10px;

cursor: pointer;

}

.suggestions-list div:hover {

background-color: #f0f0f0;

}

</style>

</head>

<body>

<div class="search-container">

<input type="text" id="search-input" placeholder="Search for books..." autocomplete="off">

<div id="suggestions" class="suggestions-list"></div>

</div>

<script src="q2.js"></script>

</body>

</html>

Js File:

$(document).ready(function() {

const books = [

{ title: "secret", author: "F. Scott Fitzgerald" },

{ title: "1984", author: "George Orwell" },

{ title: "To Kill a Mockingbird", author: "Harper Lee" },

{ title: "The Catcher in the Rye", author: "J.D. Salinger" },

{ title: "The Hobbit", author: "J.R.R. Tolkien" },

{ title: "Harry Potter and the Philosopher's Stone", author: "J.K. Rowling" },

{ title: "Pride and Prejudice", author: "Jane Austen" }

];

$('#search-input').on('input', function() {

let query = $(this).val().trim().toLowerCase();

if (query.length > 0) {

let filteredBooks = books.filter(book =>

book.title.toLowerCase().includes(query) || book.author.toLowerCase().includes(query)

);

displaySuggestions(filteredBooks);

} else {

$('#suggestions').empty().hide();

}

});

function displaySuggestions(books) {

let suggestionsBox = $('#suggestions');

suggestionsBox.empty(); // Clear any previous suggestions

if (books.length > 0) {

books.forEach(function(book) {

suggestionsBox.append(`<div class="suggestion-item">${book.title} by ${book.author}</div>`);

});

suggestionsBox.show(); // Show the suggestions box

} else {

suggestionsBox.hide(); // Hide the suggestions if no matches

}

$('.suggestion-item').on('click', function() {

$('#search-input').val($(this).text()); // Set the input to the selected suggestion

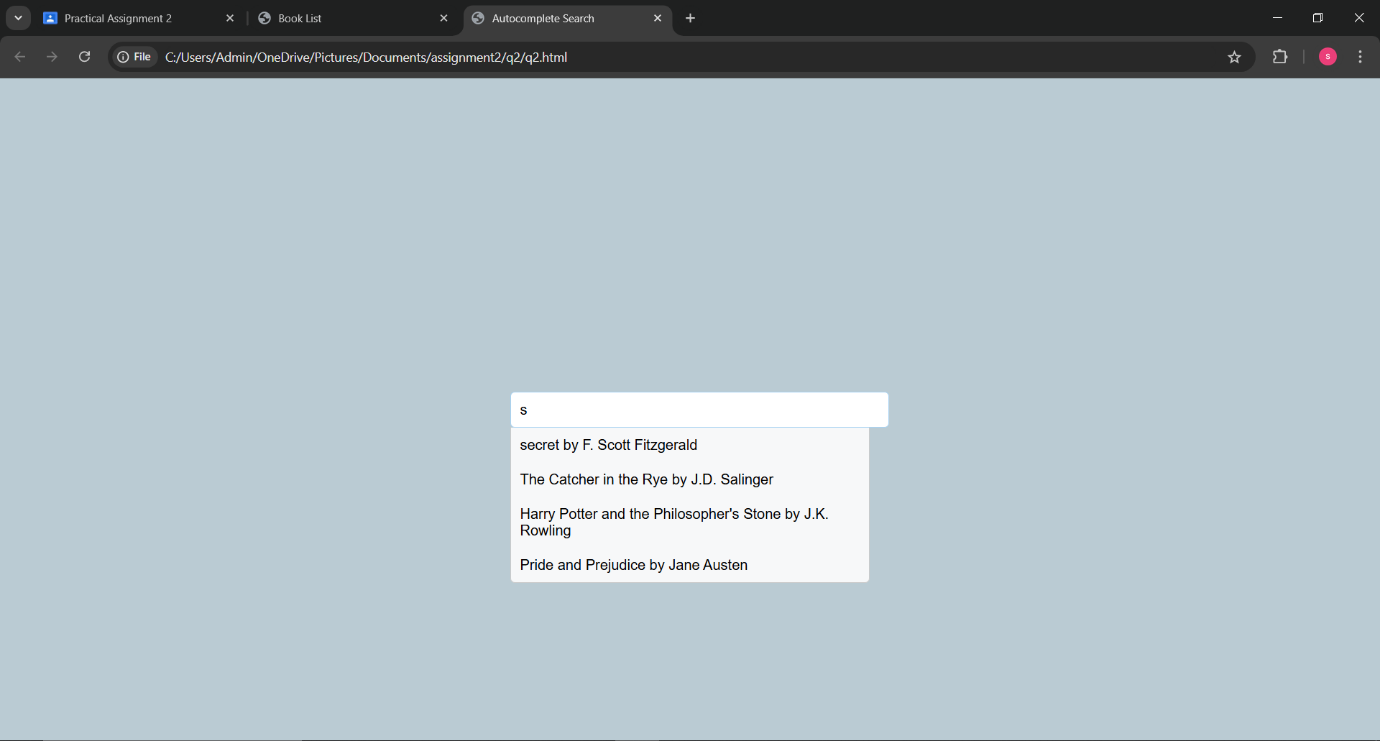
suggestionsBox.hide(); // Hide the suggestions box

});

}

});

Output:



8. Build a real-time chat application using AJAX to send and receive messages without refreshing the page. Messages should be displayed on the page as they are sent and received.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Real-time Chat Application</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f1e7e7;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

.chat-container {

width: 400px;

border: 1px solid #f4caca;

background-color: #f4d1d1;

display: flex;

flex-direction: column;

border-radius: 10px;

overflow: hidden;

}

.chat-window {

height: 400px;

border-bottom: 1px solid #ccc;

padding: 10px;

overflow-y: auto;

flex: 1;

}

.messages {

display: flex;

flex-direction: column;

}

.message {

margin-bottom: 10px;

padding: 8px 12px;

border-radius: 5px;

max-width: 75%;

}

.message.sent {

background-color: #efebed;

align-self: flex-end;

}

.message.received {

background-color: #f0f0f0;

align-self: flex-start;

}

.input-container {

display: flex;

padding: 10px;

}

#chat-input {

flex: 1;

padding: 10px;

border: 1px solid #ccc;

border-radius: 5px;

}

#send-btn {

margin-left: 10px;

padding: 10px;

background-color: #2848a7;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

}

#send-btn:hover {

background-color: #8aa1e4;

}

</style>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<div class="chat-container">

<div class="chat-window">

<div id="messages" class="messages">

</div>

</div>

<div class="input-container">

<input type="text" id="chat-input" placeholder="Type a message..." autocomplete="off">

<button id="send-btn">Send</button>

</div>

</div>

<script src="q3.js"></script>

</body>

</html>

Js File:

$(document).ready(function() {

// Simulating the backend chat functionality

let messageHistory = [];

$('#send-btn').on('click', function() {

let message = $('#chat-input').val().trim();

if (message !== '') {

sendMessage(message); // Handle sending the message

$('#chat-input').val(''); // Clear the input after sending

}

});

$('#chat-input').on('keypress', function(e) {

if (e.which === 13) { // Enter key is pressed

$('#send-btn').click(); // Trigger the send button

}

});

function sendMessage(message) {

messageHistory.push({ text: message, sender: 'user' });

displayMessage(message, 'sent');

setTimeout(function() {

let replyMessage = generateReply(message);

messageHistory.push({ text: replyMessage, sender: 'bot' });

displayMessage(replyMessage, 'received');

}, 1000);

}

function displayMessage(message, type) {

let messageElement = `<div class="message ${type}">${message}</div>`;

$('#messages').append(messageElement);

$('.chat-window').scrollTop($('.chat-window')[0].scrollHeight); // Auto-scroll to bottom

}

function generateReply(userMessage) {

let replies = [

"I'm here to help!",

"That's interesting!",

"Tell me more.",

"Got it! Anything else?",

"How can I assist you further?"

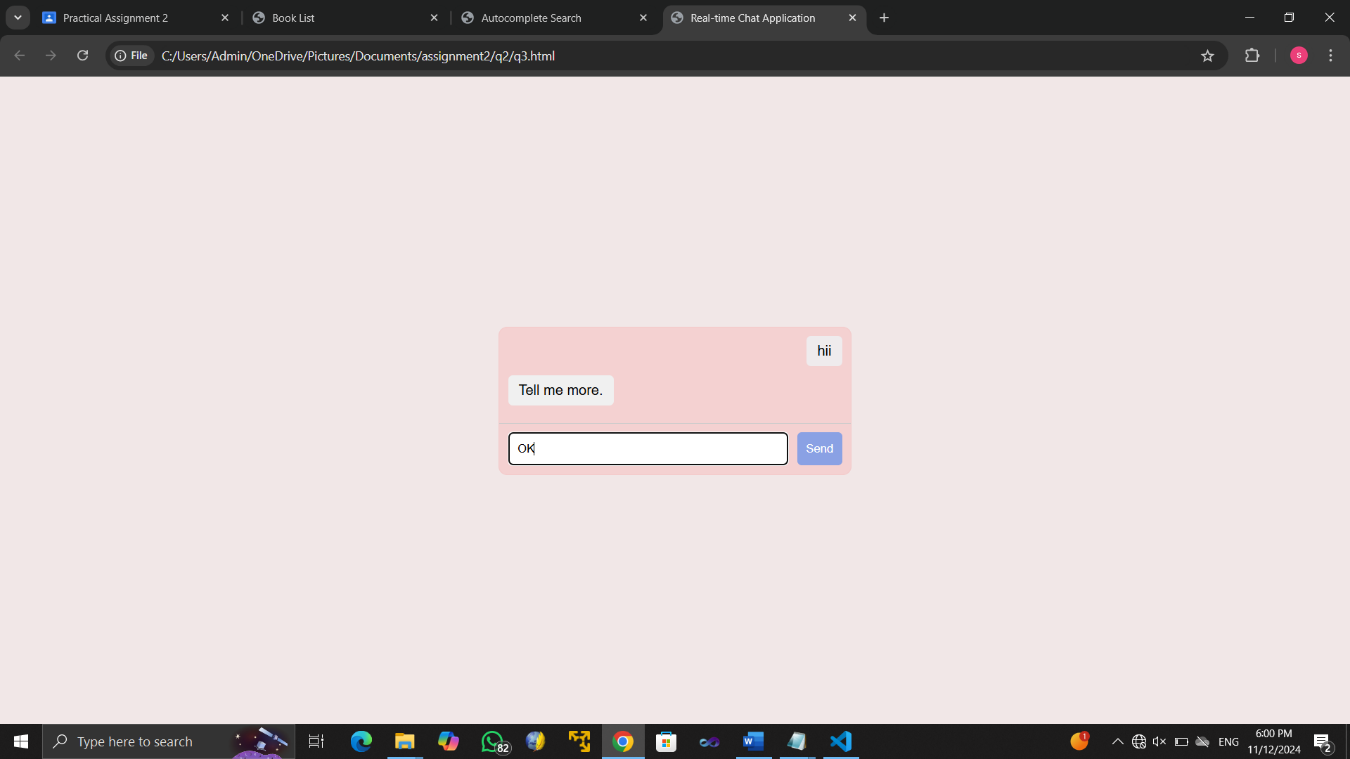
];

return replies[Math.floor(Math.random() \* replies.length)];

}

});

Output:



9. Design a “Load More” button that, when clicked, fetches additional content from an API and appends it to an existing list on the webpage. Implement pagination using AJAX.

Code;

<!DOCTYPE html>

<html lang="en">

<head>

<title>Load More with Pagination</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #6f84d9;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

.content-container {

width: 400px;

text-align: center;

}

.content-list {

list-style-type: none;

padding: 0;

}

.content-list li {

background-color: #fff;

border: 1px solid #ccc;

padding: 10px;

margin: 5px 0;

border-radius: 5px;

}

.load-more-btn {

padding: 10px 20px;

background-color: #f1f1f2;

color: rgb(24, 24, 24);

border: none;

border-radius: 5px;

cursor: pointer;

margin-top: 10px;

}

.load-more-btn:hover {

background-color: #b0cae6;

}

</style>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<div class="content-container">

<ul id="content-list" class="content-list">

</ul>

<button id="load-more-btn" class="load-more-btn">Load More</button>

</div>

<script src="q4.js"></script>

</body>

</html>

Js File:

$(document).ready(function() {

const data = [

"Item 1", "Item 2", "Item 3", "Item 4", "Item 5",

"Item 6", "Item 7", "Item 8", "Item 9", "Item 10",

"Item 11", "Item 12", "Item 13", "Item 14", "Item 15"

];

let currentPage = 1; // Current page (starts with 1)

const itemsPerPage = 5; // Number of items to load per page

function loadContent(page) {

const startIndex = (page - 1) \* itemsPerPage;

const endIndex = startIndex + itemsPerPage;

const itemsToLoad = data.slice(startIndex, endIndex);

itemsToLoad.forEach(function(item) {

$('#content-list').append(`<li>${item}</li>`);

});

if (endIndex >= data.length) {

$('#load-more-btn').hide();

}

}

loadContent(currentPage);

$('#load-more-btn').on('click', function() {

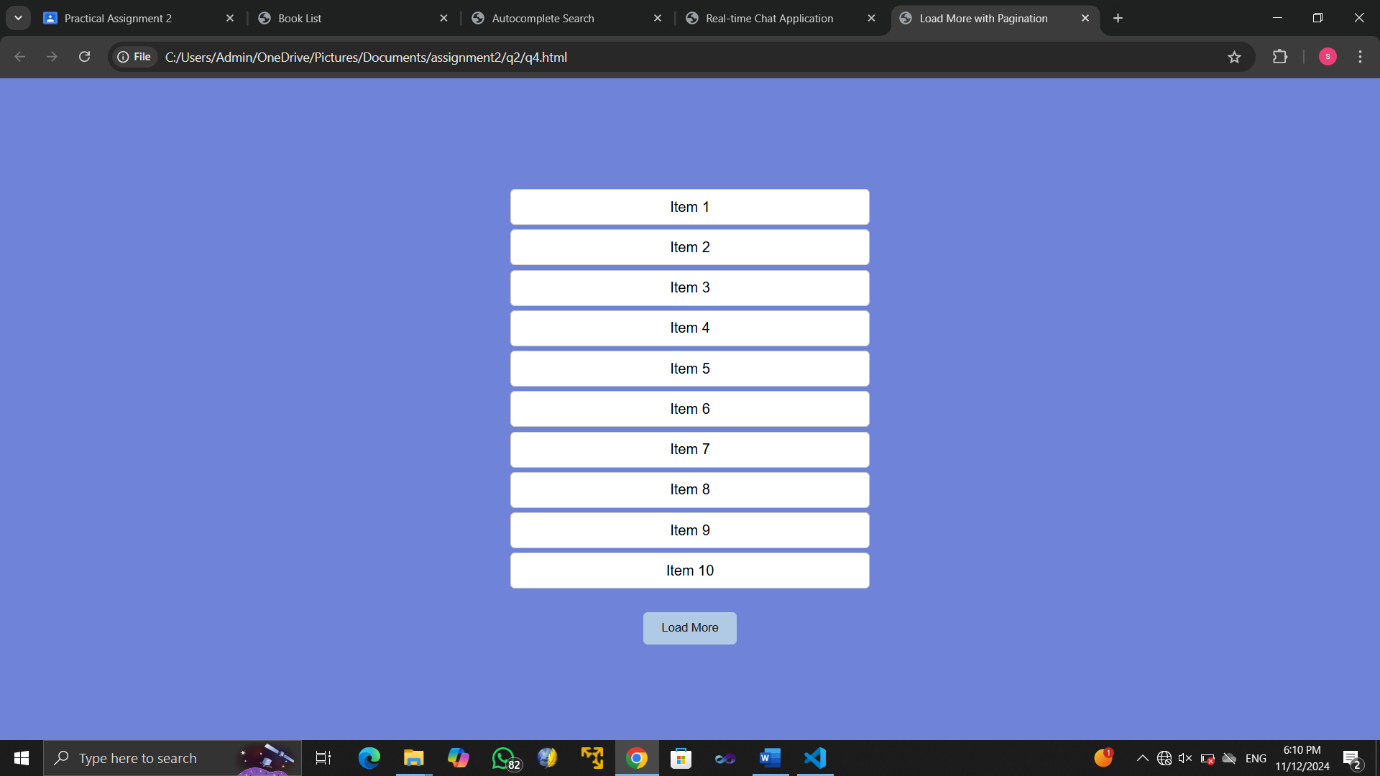
currentPage++; // Increment the page number

loadContent(currentPage); // Load the next set of items

});

});

Output:



10. Develop a weather app that allows users to enter a city name and retrieve weather data from a weather API using AJAX. Display the temperature, weather description, and an icon representing the weather condition.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Weather App</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #d8daa9;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

.weather-container {

background-color: #f2f9e4;

padding: 20px;

border-radius: 10px;

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1);

text-align: center;

width: 300px;

}

#city-input {

padding: 10px;

width: 80%;

margin-bottom: 10px;

border: 1px solid #f1bcbc;

border-radius: 5px;

}

#get-weather-btn {

padding: 10px 20px;

background-color: #dbf150;

color: rgb(10, 10, 10);

border: none;

border-radius: 5px;

cursor: pointer;

}

#weather-info {

margin-top: 20px;

}

.weather-info div {

margin: 10px 0;

}

.weather-icon {

width: 50px;

height: 50px;

}

</style>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<div class="weather-container">

<h1>Weather App</h1>

<input type="text" id="city-input" placeholder="Enter city name" autocomplete="off">

<button id="get-weather-btn">Get Weather</button>

<div id="weather-info" class="weather-info">

<!-- Weather data will be displayed here -->

</div>

</div>

<script src="q5.js"></script>

</body>

</html>

Js File:

$(document).ready(function() {

// Simulated weather data

const weatherData = {

"Surat": {

temperature: 25,

description: "Sunny",

icon: "☀️"

},

"Mumbai": {

temperature: 15,

description: "Cloudy",

icon: "☁️"

},

"Goa": {

temperature: 20,

description: "Partly Cloudy",

icon: "⛅"

},

"Delhi": {

temperature: 18,

description: "Rainy",

icon: "🌧️"

},

"Kolkata": {

temperature: 28,

description: "Sunny",

icon: "☀️"

}

};

function displayWeather(city) {

let weather = weatherData[city];

if (weather) {

$('#weather-info').html(`

<div>Temperature: ${weather.temperature}°C</div>

<div>Description: ${weather.description}</div>

<div class="weather-icon">${weather.icon}</div>

`);

} else {

$('#weather-info').html('<div>City not found. Try another one.</div>');

}

}

$('#get-weather-btn').on('click', function() {

let city = $('#city-input').val().trim();

if (city !== '') {

displayWeather(city);

}

});

$('#city-input').on('keypress', function(e) {

if (e.which === 13) {

$('#get-weather-btn').click();

}

});

});

Output:

