EX 1 RESUME BUILDER

Aim:
To create a styled HTML form for building a resume, using a combination of inline, internal, and external CSS for a well-structured and visually appealing layout.

Steps:
Step 1: Create `index.html` and open it in a text editor.

- Start with `<!DOCTYPE html>`, `<html>`, `<head>`, and `<body>` tags.

Step 2: Add meta tags and title.

- Inside `<head>`, include `<meta charset="UTF-8">`, `<meta name="viewport" content="width=device-width, initial-scale=1.0">`, and `<title>Resume Builder</title>`.

Step 3: Link external CSS.

- In `<head>`, add `<link rel="stylesheet" href="styles.css">`.

Step 4: Add internal CSS.

- Use a `<style>` tag in `<head>` to define `.section-title`, `.highlight`, and `.italic` classes.

Step 5: Create the form structure.

- In `<body>`, add an `<h1>` with inline CSS for the title and build a `<form>` with input fields for personal info, education, work experience, and skills.

Step 6: Create the external CSS file.

- Save the provided CSS code in a file named `styles.css` to style the form and its elements.

Step 7: Add a submit button.

- At the end of the form, include `<button type="submit">Submit</button>`.

HTML
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Resume Builder</title>
  <link rel="stylesheet" href="styles.css"> <!-- External CSS -->
  <style>
    /* Internal CSS */
    .section-title {
      color: #4CAF50;
      text-decoration: underline;
    }
    .highlight {
      color: #FF5733;
      font-weight: bold;
    }
    .italic {
      font-style: italic;
```

```
        }
    </style>
</head>
<body>
    <h1 style="color: #007BFF;">Resume Builder</h1> <!-- Inline CSS -->

    <form action="/submit_resume" method="post">
        <div>
            <label for="name" class="section-title">Personal Information</label><br>
            <label for="name" style="font-weight: bold;">Name:</label>
            <input type="text" id="name" name="name" required><br>
            <label for="email" class="highlight">Email:</label>
            <input type="email" id="email" name="email" required><br>
            <label for="phone" class="italic">Phone Number:</label>
            <input type="tel" id="phone" name="phone" required>
        </div>

        <div>
            <label for="education" class="section-title">Education</label><br>
            <label for="degree">Degree:</label>
            <input type="text" id="degree" name="degree" required><br>
            <label for="school" style="color: #FF5733;">School:</label>
            <input type="text" id="school" name="school" required><br>
            <label for="year" class="italic">Year of Graduation:</label>
            <input type="text" id="year" name="year" required>
        </div>

        <div>
            <label for="experience" class="section-title">Work Experience</label><br>
            <label for="job-title" style="font-weight: bold;">Job Title:</label>
            <input type="text" id="job-title" name="job-title" required><br>
            <label for="company" class="highlight">Company:</label>
            <input type="text" id="company" name="company" required><br>
            <label for="years" class="italic">Years Worked:</label>
            <input type="text" id="years" name="years" required>
        </div>

        <div>
            <label for="skills" class="section-title">Skills</label><br>
            <label for="skills" style="color: #007BFF; font-style: italic;">List your skills:</label>
            <textarea id="skills" name="skills" rows="4" cols="50" required></textarea>
        </div>

        <button type="submit">Submit</button>
    </form>
</body>
</html>
```

Save the following as styles.css

```
/* External CSS */
```

```css
body {
  font-family: Arial, sans-serif;
  background-color: #f2f2f2;
  margin: 20px;
  padding: 20px;
}

form {
  background-color: #ffffff;
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

label {
  margin-top: 10px;
  display: inline-block;
  width: 150px;
}

input, textarea {
  margin-top: 5px;
  margin-bottom: 10px;
  padding: 5px;
  width: 100%;
  border: 1px solid #ccc;
  border-radius: 5px;
}

button {
  background-color: #4CAF50;
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 5px;
  cursor: pointer;
}

button:hover {
  background-color: #45a049;
}
```

Art Gallery

Aim

To create a simple, visually appealing art gallery website that allows users to view artwork in a gallery format

Algorithm

1. Setup Project Files: Create `index.html`, `styles.css`, `scripts.js`, and add artwork images.
2. Basic HTML Structure: In `index.html`, set up HTML with references to `styles.css` and `scripts.js`.
3. Add Header: Add a `<header>` with a title inside the `<body>`.
4. Create Gallery: Add a `<main>` with a `<section id="gallery">` containing `<div class="gallery-item">` with images.
5. Add Modal: Add a `<div id="modal" class="modal">` with an `<img id="modal-img" class="modal-content">`.
6. Style with CSS: In `styles.css`, style the gallery and modal, including hover effects.
7. JavaScript Functions: In `scripts.js`, add `openModal(imageSrc)`, `closeModal()`, and `openModalOnEnter(event, imageSrc)`.
8. Link and Test: Ensure `index.html` links to `styles.css` and `scripts.js`, then open in a browser to test.

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 Art Gallery</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <h1>Art Gallery</h1>
    </header>
    <main>
        <section id="gallery">
            <div class="gallery-item" tabindex="0" onclick="openModal('art1.jpg')"
onkeypress="openModalOnEnter(event, 'art1.jpg')">
                <img src="art1.jpg" alt="Artwork 1">
            </div>
            <div class="gallery-item" tabindex="0" onclick="openModal('art2.jpg')"
onkeypress="openModalOnEnter(event, 'art2.jpg')">
                <img src="art2.jpg" alt="Artwork 2">
            </div>
            <div class="gallery-item" tabindex="0" onclick="openModal('art3.jpg')"
onkeypress="openModalOnEnter(event, 'art2.jpg')">
```

```
            <img src="art3.jpg" alt="Artwork 3">
        </div>
        <div class="gallery-item" tabindex="0" onclick="openModal('art4.jpg')"
onkeypress="openModalOnEnter(event, 'art2.jpg')">
            <img src="art4.jpg" alt="Artwork 4">
        </div>
    </section>
  </main>
  <div id="modal" class="modal" onclick="closeModal()">
    <img class="modal-content" id="modal-img">
  </div>
  <script src="scripts.js"></script>
</body>
</html>
```

scripts.js

```
function openModal(imageSrc) {
  var modal = document.getElementById("modal");
  var modalImg = document.getElementById("modal-img");
  modal.style.display = "flex";
  modalImg.src = imageSrc;
}

function closeModal() {
  var modal = document.getElementById("modal");
  modal.style.display = "none";
}
```

styles.css

```
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f4f4f4;
}

header {
  background-color: #333;
  color: #fff;
  padding: 1em 0;
  text-align: center;
}

main {
    padding: 20px;
}

#gallery {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-around;
}

.gallery-item {
    margin: 10px;
    cursor: pointer;
    outline: none;
}

.gallery-item img {
    width: 150px;
    height: auto;
    border: 2px solid #333;
    transition: transform 0.2s, border-color 0.2s;
}

.gallery-item:hover img,
.gallery-item:focus img {
    border: 2px solid #f39c12;
    transform: scale(1.05);
}

.modal {
    display: none;
    position: fixed;
    z-index: 1;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: auto;
    background-color: rgba(0, 0, 0, 0.8);
    justify-content: center;
    align-items: center;
}

.modal-content {
    max-width: 90%;
    max-height: 90%;
    margin: auto;
    display: block;
}
```

Ex.3 Build a responsive web application for shopping cart with registration, login, catalog and cart pages using CSS features, flex and grid.

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>User Registration and Login</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div id="container">
    <div id="register">
      <h2>Register</h2>
      <form id="registerForm">
        <label for="regUsername">Username:</label>
        <input type="text" id="regUsername" required><br>
        <label for="regPassword">Password:</label>
        <input type="password" id="regPassword" required><br>
        <button type="submit">Register</button>
      </form>
    </div>

    <div id="login" style="display:none;">
      <h2>Login</h2>
      <form id="loginForm">
        <label for="loginUsername">Username:</label>
        <input type="text" id="loginUsername" required><br>
        <label for="loginPassword">Password:</label>
        <input type="password" id="loginPassword" required><br>
        <button type="submit">Login</button>
      </form>
    </div>

    <div id="catalog" style="display:none;">
      <h2>Catalog</h2>
      <div id="items">
        <div>Item 1 - $10 <button onclick="addToCart('Item 1', 10)">Add to Cart</button></div>
        <div>Item 2 - $20 <button onclick="addToCart('Item 2', 20)">Add to Cart</button></div>
        <div>Item 3 - $30 <button onclick="addToCart('Item 3', 30)">Add to Cart</button></div>
      </div>
      <button onclick="viewCart()">View Cart</button>
    </div>

    <div id="cart" style="display:none;">
      <h2>Cart</h2>
      <div id="cartItems">
        <!-- Cart items will be added here dynamically -->
```

```
            </div>
            <button onclick="viewCatalog()">Back to Catalog</button>
        </div>
    </div>
    <script src="script.js"></script>
</body>
</html>
```

styles.css

```
body {
    font-family: Arial, sans-serif;
}

#container {
    width: 300px;
    margin: auto;
}

h2 {
    text-align: center;
}

form {
    display: flex;
    flex-direction: column;
}

label, input, button {
    margin: 5px 0;
}

#items div, #cartItems div {
    margin: 10px 0;
    display: flex;
    justify-content: space-between;
}
```

script.js

```
let users = [];
let currentUser = null;
let cart = [];

document.getElementById('registerForm').addEventListener('submit', function(event) {
    event.preventDefault();
    const username = document.getElementById('regUsername').value;
    const password = document.getElementById('regPassword').value;

    console.log('Register form submitted');
    console.log('Username:', username, 'Password:', password);
    if (username && password) {
        if (!users.find(user => user.username === username)) {
            users.push({ username, password });
            console.log('Users array:', users);
            alert('Registration successful');
            document.getElementById('register').style.display = 'none';
            document.getElementById('login').style.display = 'block';
        } else {
            alert('Username already exists');
        }
    } else {
        alert('Please fill in both fields');
    }
});

document.getElementById('loginForm').addEventListener('submit', function(event) {
    event.preventDefault();
    const username = document.getElementById('loginUsername').value;
    const password = document.getElementById('loginPassword').value;

    console.log('Login form submitted');
    console.log('Username:', username, 'Password:', password);

    const user = users.find(u => u.username === username && u.password === password);

    if (user) {
        currentUser = user;
        console.log('Logged in user:', currentUser);
        alert('Login successful');
        document.getElementById('login').style.display = 'none';
        document.getElementById('catalog').style.display = 'block';
    } else {
        alert('Invalid username or password');
    }
});

function addToCart(itemName, itemPrice) {
    console.log('Adding to cart:', itemName, itemPrice);
    cart.push({ name: itemName, price: itemPrice });
    console.log('Cart:', cart);
    alert('Item added to cart');
}

function viewCart() {
    console.log('Viewing cart');
    document.getElementById('catalog').style.display = 'none';
    document.getElementById('cart').style.display = 'block';
    const cartItemsDiv = document.getElementById('cartItems');
    cartItemsDiv.innerHTML = '';
    cart.forEach(item => {
        const cartItemDiv = document.createElement('div');
        cartItemDiv.innerHTML = `${item.name} - $${item.price}`;
        cartItemsDiv.appendChild(cartItemDiv);
    });
}

function viewCatalog() {
    console.log('Viewing catalog');
    document.getElementById('cart').style.display = 'none';
    document.getElementById('catalog').style.display = 'block';
}
```

Study the following on the internet

- **`ng-app`**
- **`ng-controller`**
- **`ng-submit`**
- **`ng-model`**
- **`ng-show`**
- **`ng-disabled`**

EX NO 4

Index.html

```
<!DOCTYPE html>
<html ng-app="registrationApp">
<head>
    <meta charset="UTF-8">
    <title>User Registration</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
    <script>
        function validatePasswordMismatch() {
            var password = document.getElementById('password').value;
            var confirmPassword = document.getElementById('confirmPassword').value;
            var passwordError = document.getElementById('passwordError');

            if (password !== confirmPassword) {
                passwordError.style.display = 'block';
                return false;
            } else {
                passwordError.style.display = 'none';
                return true;
            }
        }
    </script>
</head>
<body ng-controller="RegistrationController">
    <div class="container">
        <h2>User Registration Form</h2>
        <form name="registrationForm" ng-submit="submitForm($event)" novalidate>
            <div class="form-group">
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" class="form-control" ng-
model="user.username" required>
                <div ng-show="registrationForm.username.$touched &&
registrationForm.username.$invalid" class="text-danger">
                    Username is required.
                </div>
            </div>
            <div class="form-group">
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" class="form-control" ng-
model="user.email" required>
                <div ng-show="registrationForm.email.$touched && registrationForm.email.$invalid"
class="text-danger">
                    Valid email is required.
                </div>
            </div>
            <div class="form-group">
                <label for="password">Password:</label>
                <input type="password" id="password" name="password" class="form-control" ng-
model="user.password" required>
                <div ng-show="registrationForm.password.$touched &&
registrationForm.password.$invalid" class="text-danger">
                    Password is required.
                </div>
            </div>
            <div class="form-group">
                <label for="confirmPassword">Confirm Password:</label>
                <input type="password" id="confirmPassword" name="confirmPassword"
class="form-control" required oninput="validatePasswordMismatch()">
                <div id="passwordError" class="text-danger" style="display: none;">Passwords must
match.</div>
            </div>
            <button type="submit" class="btn btn-primary" ng-
disabled="registrationForm.$invalid">Register</button>
        </form>
    </div>

    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
    <script src="app.js"></script>
</body>
</html>
```

app.js

```
var app = angular.module('registrationApp', []);

app.controller('RegistrationController', function($scope) {
    $scope.user = {};

    $scope.submitForm = function(event) {
        if ($scope.registrationForm.$valid && validatePasswordMismatch()) {
            alert("Form submitted successfully!");
            console.log($scope.user);
        } else {
            alert("Please correct the errors in the form.");
            event.preventDefault();
        }
    };
});
```

EX.5 Develop a quiz application with AngularJS, featuring multiple-choice questions and scoring systems

Index.html

```html
<!DOCTYPE html>
<html ng-app="quizApp">
<head>
  <meta charset="UTF-8">
  <title>Quiz Application</title>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
  <script src="app.js"></script>
  <script src="quizController.js"></script>
  <script src="quizService.js"></script>
  <link rel="stylesheet" href="styles.css">
</head>
<body ng-controller="QuizController">

  <div class="quiz-container" ng-show="!quizCompleted">
    <h1>{{ title }}</h1>
    <div class="question">
      <h3>{{ currentQuestion.text }}</h3>
      <div class="options" ng-repeat="option in currentQuestion.options">
        <input type="radio" ng-model="currentQuestion.userAnswer" ng-value="option"> {{ option }}
      </div>
    </div>
    <button ng-click="nextQuestion()" ng-disabled="!currentQuestion.userAnswer">Next</button>
  </div>

  <div class="result-container" ng-show="quizCompleted">
    <h1>Quiz Completed!</h1>
    <h2>Your score: {{ score }}/{{ questions.length }}</h2>
  </div>

</body>
</html>
```

app.js

```javascript
angular.module('quizApp', []);
```

quizController.js

```javascript
angular.module('quizApp')
  .controller('QuizController', ['$scope', 'QuizService', function($scope, QuizService) {
    $scope.title = "Quiz Application";
    $scope.questions = QuizService.getQuestions();
    $scope.score = 0;
    $scope.quizCompleted = false;
    $scope.currentQuestionIndex = 0;
    $scope.currentQuestion = $scope.questions[$scope.currentQuestionIndex];

    $scope.nextQuestion = function() {
      if ($scope.currentQuestion.userAnswer === $scope.currentQuestion.correctAnswer) {
        $scope.score++;
      }
      $scope.currentQuestionIndex++;
      if ($scope.currentQuestionIndex < $scope.questions.length) {
        $scope.currentQuestion = $scope.questions[$scope.currentQuestionIndex];
      } else {
        $scope.quizCompleted = true;
      }
    };
  }]);
```

quizService.js

```
angular.module('quizApp')
  .service('QuizService', function() {
    var questions = [
      {
        text: 'What is the capital of France?',
        options: ['Paris', 'London', 'Rome', 'Berlin'],
        correctAnswer: 'Paris'
      },
      {
        text: 'Which planet is known as the Red Planet?',
        options: ['Earth', 'Mars', 'Jupiter', 'Saturn'],
        correctAnswer: 'Mars'
      },
      {
        text: 'What is the largest ocean on Earth?',
        options: ['Atlantic Ocean', 'Indian Ocean', 'Arctic Ocean', 'Pacific Ocean'],
        correctAnswer: 'Pacific Ocean'
      }
    ];

    this.getQuestions = function() {
      return questions;
    };
  });
```

styles.css

```
body {
  font-family: Arial, sans-serif;
  background-color: #f4f4f4;
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

.quiz-container, .result-container {
  background-color: #fff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

h1, h2 {
  text-align: center;
}

.question {
  margin-bottom: 20px;
}

.options {
  margin-bottom: 10px;
}

button {
  display: block;
  width: 100%;
  padding: 10px;
  background-color: #28a745;
  color: #fff;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

button:hover {
  background-color: #218838;
}

button:disabled {
  background-color: #cccccc;
  cursor: not-allowed;
}
```

Fetch weather data from an API and display it dynamically in a user-friendly interface using AngularJS

Index.html

```
<!DOCTYPE html>
<html lang="en" ng-app="weatherApp">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Weather App</title>
   <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
   <link rel="stylesheet" href="styles.css">
</head>
<body ng-controller="WeatherController">

   <div class="weather-card">
      <h2>Weather in {{city}}</h2>
      <input type="text" ng-model="city" placeholder="Enter city" />
      <button ng-click="getWeather()">Get Weather</button>

      <div ng-if="weatherData">
         <h3>{{weatherData.name}}, {{weatherData.sys.country}}</h3>
         <p>Temperature: {{weatherData.main.temp}} °C</p>
         <p>Weather: {{weatherData.weather[0].description}}</p>
      </div>

      <div ng-if="error">
         <p style="color: red;">{{error}}</p>
      </div>
   </div>

   <script src="app.js"></script>
</body>
</html>
```

app.js

```
var app = angular.module('weatherApp', []);

app.controller('WeatherController', function($scope, $http) {
   $scope.city = 'London'; // Default city

   $scope.getWeather = function() {
      const apiKey = 'Your  API KEY'; // Replace with your OpenWeatherMap API key from
https://openweathermap.org/
      const apiUrl =
`https://api.openweathermap.org/data/2.5/weather?q=${$scope.city}&units=metric&appid=${apiKey
}`;

      $http.get(apiUrl).then(function(response) {
         $scope.weatherData = response.data;
         $scope.error = null; // Clear any previous errors
      }, function(error) {
         $scope.error = 'Error fetching weather data. Please try again.';
         $scope.weatherData = null; // Clear previous data
```

```
        });
    };

    // Fetch weather data for the default city on load
    $scope.getWeather();
});
```

Styles.css

```
body {
    font-family: Arial, sans-serif;
    margin: 20px;
}

.weather-card {
    border: 1px solid #ccc;
    padding: 20px;
    border-radius: 5px;
    width: 300px;
    margin: auto;
}
```