Project Name:
MoodTunes - Mood-Based Playlist Generator.

Description:
Mood Tunes is an interactive web application designed to create personalized playlists based on the user's current mood. By simply describing how they feel in a text box, users can receive a curated list of songs that match their emotional state. Whether they're feeling happy, sad, angry, or calm, Mood Tunes analyzes the input and suggests relevant tracks, complete with links to play them on YouTube.

The application features a clean and user-friendly interface, making it easy for anyone to use. It leverages basic natural language processing to detect keywords in the user's input and generates a playlist accordingly. Mood Tunes is perfect for music lovers who want to enhance their mood or find songs that resonate with their current emotions.

Coding Languages Used:
1. HTML: Used for structuring the content and layout of the web application.
2. CSS: Used for styling the application, including fonts, colors, and responsive design.
3. JavaScript: Used for implementing the logic, such as mood analysis, playlist generation, and dynamic content display.

Techniques Used:
1. Natural Language Processing (NLP) - Basic:
 - The application uses simple keyword matching (`includes`) to analyze the user's input and determine their mood (e.g., happy, sad, angry, calm).

2. Dynamic Content Rendering:
 - JavaScript dynamically generates and displays the playlist based on the user's mood. The playlist section is hidden by default and only appears after the user submits their input.

3. Responsive Design:
 - CSS Flex box is used to ensure the application is centered and responsive across different screen sizes.

4. Event Handling:
 - The `on-click` event is used to trigger the mood analysis and playlist generation when the user clicks the "Generate Playlist" button.

5. External Integration:
 - The application integrates with YouTube by providing direct links to songs using their `video Id`. This allows users to play the suggested songs directly on YouTube.

6. Conditional Rendering:
 - If no songs are found for a specific mood, a fallback message ("No songs found for your mood") is displayed to the user.

7. Modular Code Structure:
 - The JavaScript code is organized into reusable functions (`analyze Mood`, `generate Playlist`, `display Playlist`) for better readability and maintainability.

Example of Key Techniques in Action:

- Mood Analysis:

```javascript
function analyzeMoodText(text) {
  if (text.toLowerCase().includes('happy')) return 'happy';
  if (text.toLowerCase().includes('sad')) return 'sad';
  if (text.toLowerCase().includes('angry')) return 'angry';
  return 'calm';
}
```

- **Dynamic Playlist Display**:

```javascript
function displayPlaylist(playlist) {
  const playlistSection = document.getElementById('playlistSection');
  const playlistDiv = document.getElementById('playlist');
  playlistDiv.innerHTML = '';

  if (playlist.length === 0) {
    playlistDiv.innerHTML = '<p>No songs found for your mood.</p>';
    playlistSection.style.display = 'block';
    return;
  }

  playlist.forEach(song => {
    const songItem = document.createElement('div');
    songItem.className = 'playlist-item';
    songItem.innerHTML = `
      <p>
        <strong>${song.title}</strong><br>
        ${song.artist}<br>
        <a href="https://www.youtube.com/watch?v=${song.videoId}" target="_blank">Play on YouTube</a>
      </p>
    `;
    playlistDiv.appendChild(songItem);
  });

  playlistSection.style.display = 'block';
}
```

Code:-

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

```
<style>
 /* General Styles */
  body {
    font-family: "STsong";
    background: linear-gradient(135deg, #ffcccb, #ffffff);
    color: #333;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    flex-direction: column;
  }

  .container {
    background: rgba(255, 255, 255, 0.1);
    backdrop-filter: blur(10px);
    border-radius: 15px;
    padding: 2rem;
    width: 90%;
    max-width: 500px;
    text-align: center;
    box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
    border: 1px solid rgba(255, 255, 255, 0.1);
  }

  h1 {
    margin-bottom: 1.5rem;
    font-size: 2rem;
    font-weight: 600;
    color: #333;
  }

  textarea {
    width: 100%;
    height: 100px;
    padding: 10px;
    border: 1px solid rgba(255, 255, 255, 0.3);
    border-radius: 10px;
    margin-bottom: 1rem;
    font-size: 16px;
    background: rgba(255, 255, 255, 0.1);
    color: #333;
    resize: none;
    transition: border-color 0.3s ease;
  }

  textarea::placeholder {
```

```
  color: rgba(255, 255, 255, 0.7);
}

textarea:focus {
  outline: none;
  border-color: #6c5ce7;
}

button {
  background-color: #6c5ce7;
  color: #333;
  border: none;
  padding: 12px 24px;
  border-radius: 10px;
  cursor: pointer;
  font-size: 16px;
  font-weight: 500;
  transition: background-color 0.3s ease, transform 0.2s ease;
}

button:hover {
  background-color: #5a4fcf;
  transform: translateY(-2px);
}

button:active {
  transform: translateY(0);
}

.playlist {
  margin-top: 2rem;
  text-align: left;
}

.playlist h2 {
  margin-bottom: 1rem;
  font-size: 1.5rem;
  font-weight: 600;
  color: #333;
}

.playlist-item {
  display: flex;
  align-items: center;
  margin-bottom: 1rem;
  padding: 10px;
  background: rgba(255, 255, 255, 0.1);
  border-radius: 10px;
  transition: transform 0.3s ease, background 0.3s ease;
```

```
}

.playlist-item:hover {
  background: rgba(255, 255, 255, 0.2);
  transform: translateX(10px);
}

.playlist-item img {
  width: 50px;
  height: 50px;
  border-radius: 5px;
  margin-right: 1rem;
}

.playlist-item p {
  margin: 0;
  font-size: 14px;
  color: #333;
}

.playlist-item a {
  display: inline-flex;
  align-items: center;
  background-color: #ff0000; /* YouTube red */
  color: #333;
  padding: 8px 12px;
  border-radius: 5px;
  text-decoration: none;
  font-weight: 500;
  transition: background-color 0.3s ease, transform 0.2s ease;
}

.playlist-item a:hover {
  background-color: #cc0000; /* Darker red on hover */
  transform: translateY(-2px);
}

.playlist-item a:active {
  transform: translateY(0);
}

.playlist-item a i {
  margin-right: 8px;
}

.saved-moods {
  margin-top: 2rem;
  text-align: left;
}
```

```
    .saved-moods h2 {
      margin-bottom: 1rem;
      font-size: 1.5rem;
      font-weight: 600;
      color: #333;
    }

    .saved-moods ul {
      list-style: none;
      padding: 0;
    }

    .saved-moods li {
      display: flex;
      justify-content: space-between;
      align-items: center;
      padding: 10px;
      background: rgba(255, 255, 255, 0.1);
      border-radius: 10px;
      margin-bottom: 0.5rem;
      transition: transform 0.3s ease, background 0.3s ease;
    }

    .saved-moods li:hover {
      background: rgba(255, 255, 255, 0.2);
      transform: translateX(10px);
    }

    .saved-moods li button {
      background: #ff4757;
      color: #333;
      border: none;
      padding: 5px 10px;
      border-radius: 5px;
      cursor: pointer;
      font-size: 12px;
      transition: background 0.3s ease;
    }

    .saved-moods li button:hover {
      background: #ff6b81;
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Mood-Based Playlist Generator</h1>
    <textarea id="moodInput" placeholder="How are you feeling today?"></textarea>
```

```
  <button onclick="analyzeMood()">Generate Playlist</button>

  <div class="playlist" id="playlistSection" style="display: none;">
    <h2>Your Playlist</h2>
    <div id="playlist"></div>
  </div>

  <!-- Display saved moods -->
  <div class="saved-moods" id="savedMoodsSection" style="display: none;">
    <h2>Saved Moods</h2>
    <ul id="savedMoodsList"></ul>
  </div>
</div>

<script>
  // Function to analyze mood and generate playlist
  function analyzeMood() {
    const moodInput = document.getElementById('moodInput').value;
    if (!moodInput.trim()) {
      alert('Please describe your mood!');
      return;
    }

    const mood = analyzeMoodText(moodInput);
    const playlist = generatePlaylist(mood);
    displayPlaylist(playlist);

    // Save the mood to localStorage
    saveMood(mood);
  }

  // Function to analyze mood text
  function analyzeMoodText(text) {
    const moodKeywords = {
      happy: ['happy', 'joyful', 'excited', 'cheerful'],
      sad: ['sad', 'depressed', 'unhappy', 'heartbroken'],
      angry: ['angry', 'frustrated', 'annoyed', 'irritated'],
      calm: ['calm', 'peaceful', 'relaxed', 'serene'],
      energetic: ['energetic', 'pumped', 'active', 'lively'],
      romantic: ['romantic', 'loving', 'affectionate', 'passionate'],
      nostalgic: ['nostalgic', 'sentimental', 'wistful', 'melancholic'],
      focused: ['focused', 'determined', 'motivated', 'productive'],
    };

    for (const [mood, keywords] of Object.entries(moodKeywords)) {
      if (keywords.some(keyword => text.toLowerCase().includes(keyword))) {
        return mood;
      }
    }
```

```
  return 'calm'; // Default mood
}

// Function to generate playlist based on mood
function generatePlaylist(mood) {
  const playlists = {
    happy: [
      { title: 'Uptown Funk', artist: 'Mark Ronson ft. Bruno Mars', videoId: 'OPf0YbXqDm0' },
      { title: 'Can\'t Stop the Feeling!', artist: 'Justin Timberlake', videoId: 'ru0K8uYEZWw' },
    ],
    sad: [
      { title: 'Someone Like You', artist: 'Adele', videoId: 'hLQl3WQQoQ0' },
      { title: 'Fix You', artist: 'Coldplay', videoId: 'k4V3Mo61fJM' },
    ],
    angry: [
      { title: 'Break Stuff', artist: 'Limp Bizkit', videoId: 'ZpUYjpKg9KY' },
      { title: 'Boulevard of Broken Dreams', artist: 'Green Day', videoId: 'Soa3gO7tL-c' },
    ],
    calm: [
      { title: 'Weightless', artist: 'Marconi Union', videoId: 'UfcAVejslrU' },
      { title: 'Clair de Lune', artist: 'Claude Debussy', videoId: 'CvFH_6DNRCY' },
    ],
    energetic: [
      { title: 'Eye of the Tiger', artist: 'Survivor', videoId: 'btPJPFnesV4' },
      { title: 'Lose Yourself', artist: 'Eminem', videoId: '_Yhyp-_hX2s' },
    ],
    romantic: [
      { title: 'Thinking Out Loud', artist: 'Ed Sheeran', videoId: 'lp-EO5I60KA' },
      { title: 'All of Me', artist: 'John Legend', videoId: '450p7goxZqg' },
    ],
    nostalgic: [
      { title: 'Bohemian Rhapsody', artist: 'Queen', videoId: 'fJ9rUzIMcZQ' },
      { title: 'Hotel California', artist: 'Eagles', videoId: 'BciS5krYL80' },
    ],
    focused: [
      { title: 'River Flows in You', artist: 'Yiruma', videoId: '7maJOI3QMu0' },
      { title: 'Weightless', artist: 'Marconi Union', videoId: 'UfcAVejslrU' },
    ],
  };
  return playlists[mood] || [];
}

// Function to display playlist
function displayPlaylist(playlist) {
  const playlistSection = document.getElementById('playlistSection');
  const playlistDiv = document.getElementById('playlist');
  playlistDiv.innerHTML = '';

  if (playlist.length === 0) {
```

```
      playlistDiv.innerHTML = '<p>No songs found for your mood.</p>';
      playlistSection.style.display = 'block';
      return;
    }

    playlist.forEach(song => {
      const songItem = document.createElement('div');
      songItem.className = 'playlist-item';
      songItem.innerHTML = `
        <p>
          <strong>${song.title}</strong><br>
          ${song.artist}<br>
          <a href="https://www.youtube.com/watch?v=${song.videoId}" target="_blank">Play on YouTube</a>
        </p>
      `;
      playlistDiv.appendChild(songItem);
    });

    playlistSection.style.display = 'block';
  }

  // Function to save mood to localStorage
  function saveMood(mood) {
    let savedMoods = JSON.parse(localStorage.getItem('savedMoods')) || [];
    savedMoods.push({ mood: mood, timestamp: new Date().toLocaleString() });
    localStorage.setItem('savedMoods', JSON.stringify(savedMoods));

    // Display saved moods
    displaySavedMoods();
  }

  // Function to display saved moods
  function displaySavedMoods() {
    const savedMoodsSection = document.getElementById('savedMoodsSection');
    const savedMoodsList = document.getElementById('savedMoodsList');
    savedMoodsList.innerHTML = '';

    const savedMoods = JSON.parse(localStorage.getItem('savedMoods')) || [];
    if (savedMoods.length === 0) {
      savedMoodsList.innerHTML = '<p>No moods saved yet.</p>';
    } else {
      savedMoods.forEach((entry, index) => {
        const moodItem = document.createElement('li');
        moodItem.innerHTML = `
          <strong>${entry.mood}</strong> - ${entry.timestamp}
          <button onclick="deleteMood(${index})">Delete</button>
        `;
        savedMoodsList.appendChild(moodItem);
```

```
      });
    }

    savedMoodsSection.style.display = 'block';
  }

  // Function to delete a saved mood
  function deleteMood(index) {
    let savedMoods = JSON.parse(localStorage.getItem('savedMoods')) || [];
    savedMoods.splice(index, 1); // Remove the mood at the specified index
    localStorage.setItem('savedMoods', JSON.stringify(savedMoods));

    // Refresh the displayed moods
    displaySavedMoods();
  }

  // Display saved moods when the page loads
  window.onload = displaySavedMoods;
 </script>
</body>
</html>
```

output:-