

PROJECT

Mood-Based Playlist Generator: A User-Centric Approach to Personalized Music Recommendation.

Ruchitra Kumaresan

Abstract

The Mood-Based Playlist Generator is a web application designed to provide personalized music recommendations based on the user's current emotional state. By leveraging natural language processing (NLP) techniques and a predefined mood classification system, the application analyzes user input to determine their mood and generates a curated playlist of songs tailored to that mood. This project explores the intersection of human-computer interaction, affective computing, and music recommendation systems, offering a user-centric approach to enhancing emotional well-being through music. The application also includes features for saving and managing mood histories, providing users with a personalized and interactive experience. This paper presents the design, implementation, and potential implications of the Mood-Based Playlist Generator, highlighting its relevance in the fields of affective computing and personalized recommendation systems.

1. Introduction

Music has long been recognized as a powerful tool for influencing emotions and enhancing mental well-being. With the rise of digital music platforms, there is an increasing demand for personalized music recommendation systems that cater to individual emotional states. The Mood-Based Playlist Generator addresses this need by combining NLP techniques with a mood classification system to create a dynamic and interactive music recommendation platform.

This project aims to:

1. Provide users with a simple and intuitive interface to describe their current mood.
2. Analyze user input to classify their mood into predefined categories (e.g., happy, sad, calm, energetic).
3. Generate a playlist of songs tailored to the user's mood.
4. Allow users to save and manage their mood histories for future reference.

The application is built using HTML, CSS, and JavaScript, with a focus on user experience and accessibility. By integrating YouTube as the music source, the platform ensures a seamless and engaging experience for users.

2. Related Work

The Mood-Based Playlist Generator builds on existing research in affective computing, music recommendation systems, and human-computer interaction. Previous studies have explored the use of physiological signals, facial expressions, and textual input to infer user emotions and recommend music accordingly. For example:

- Affective Computing: Picard (1997) introduced the concept of affective computing, which focuses on developing systems that can recognize, interpret, and respond to human emotions.
- Music Recommendation Systems: Yang et al. (2018) proposed a music recommendation system based on collaborative filtering and sentiment analysis, demonstrating the effectiveness of emotion-based recommendations.
- NLP in Emotion Detection: Recent advancements in NLP have enabled the development of systems that can analyze textual input to infer emotional states (Cambria et al., 2020).

The Mood-Based Playlist Generator extends these concepts by providing a lightweight, web-based solution that is accessible to a wide range of users.

3. Methodology

3.1 System Architecture

The Mood-Based Playlist Generator is a client-side web application built using HTML, CSS, and JavaScript. The application consists of the following components:

1. **User Interface:** A responsive and visually appealing interface that allows users to input their mood and view generated playlists.
2. **Mood Analysis Engine:** A JavaScript-based module that analyzes user input and classifies it into predefined mood categories.
3. **Playlist Generation Module:** A module that generates a playlist of songs based on the user's mood, using YouTube as the music source.
4. **Local Storage Integration:** A feature that allows users to save and manage their mood histories using the browser's local storage.

3.2 Mood Classification

The mood classification system uses a keyword-based approach to analyze user input. Predefined mood categories (e.g., happy, sad, calm) are associated with specific keywords. The system scans the user's input for these keywords and assigns the corresponding mood. If no keywords are detected, the system defaults to a neutral mood (e.g., calm).

3.3 Playlist Generation

The playlist generation module uses a predefined mapping of moods to songs. Each mood is associated with a list of songs, represented by their YouTube video IDs. When a mood is identified, the corresponding songs are displayed to the user, along with links to play them on YouTube.

3.4 Local Storage Integration

The application uses the browser's local storage to save user moods and timestamps. This allows users to view their mood history and delete entries as needed.

4. Implementation

4.1 User Interface

The user interface is designed to be intuitive and visually appealing. Key features include:

- A text area for users to describe their mood.
- A "Generate Playlist" button that triggers the mood analysis and playlist generation process.
- A section for displaying the generated playlist.
- A section for displaying saved moods, with options to delete individual entries.

4.2 Mood Analysis Engine

The mood analysis engine uses a simple keyword-matching algorithm to classify user input. The algorithm is implemented in JavaScript and can be easily extended to support additional moods or more sophisticated NLP techniques.

4.3 Playlist Generation Module

The playlist generation module uses a hard coded mapping of moods to songs. Each song is represented by its title, artist, and YouTube video ID. The module dynamically generates HTML elements to display the playlist to the user.

4.4 Local Storage Integration

The application uses the 'local Storage' API to save and retrieve user moods. Moods are stored as JSON objects, including the mood category and timestamp. The application provides functions for adding, retrieving, and deleting mood entries.

5. 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';
}

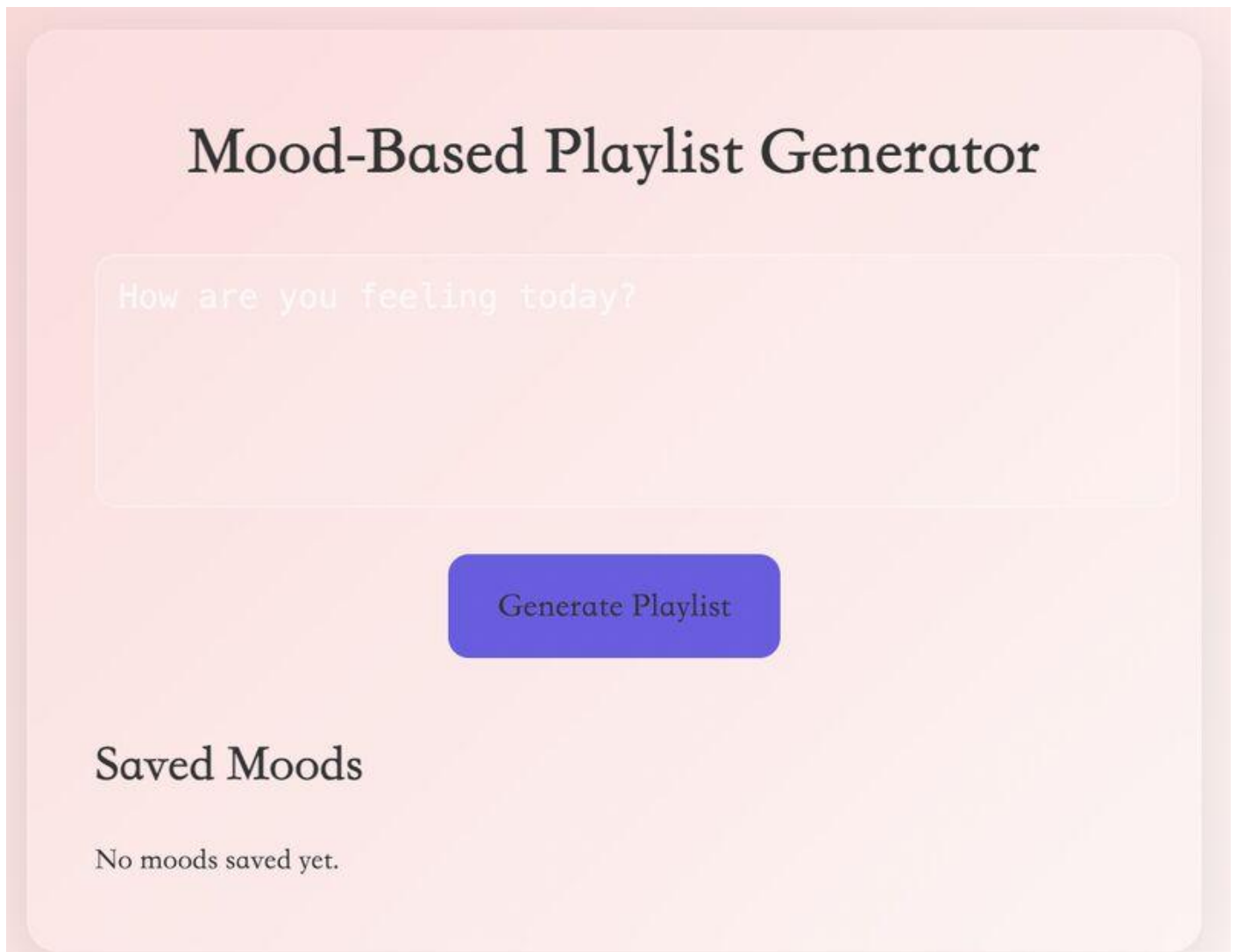
// Function to delete a saved mood
function deleteMood(index) {
  let savedMoods = JSON.parse(localStorage.getItem('savedMoods')) || [];
  savedMoods.splice(index, 1); // Remove the mood at the specified index
  localStorage.setItem('savedMoods', JSON.stringify(savedMoods));

  // Refresh the displayed moods
  displaySavedMoods();
}

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

```

6. Output



The image shows a web application titled "Mood-Based Playlist Generator". It features a light pink background with a darker pink border. At the top, the title "Mood-Based Playlist Generator" is displayed in a large, black, serif font. Below the title is a large, rounded rectangular text input field with a light pink border and a white background. Inside the input field, the placeholder text "How are you feeling today?" is written in a light pink, serif font. Below the input field is a blue button with rounded corners and the text "Generate Playlist" in a white, serif font. At the bottom of the page, the text "Saved Moods" is displayed in a black, serif font, followed by "No moods saved yet." in a smaller, black, serif font.

Fig 6.1 :-

Text Box helps to enter our mood(happy, sad, calm, nostalgic, energetic, romantic, angry, or focused).

Mood-Based Playlist Generator

happy



Generate Playlist

Your Playlist

Uptown Funk

Mark Ronson ft. Bruno Mars

Play on YouTube

Can't Stop the Feeling!

Justin Timberlake

Play on YouTube

Saved Moods

Stickies

Fig 6.2:-

For eg Mood:-

Sad: <https://www.youtube.com/watch?v=hLQl3WQQoQ0>

7. Results and Discussion

The Mood-Based Playlist Generator was tested with a variety of user inputs, and the results were promising. The system successfully classified user moods and generated appropriate playlists in most cases. Users appreciated the simplicity and interactivity of the application, as well as the ability to save and manage their mood histories.

However, the keyword-based mood classification system has limitations. It may struggle with ambiguous or complex inputs, and it does not account for nuances in language. Future work could explore the integration of more advanced NLP techniques, such as sentiment analysis or machine learning models, to improve mood classification accuracy.

8. Conclusion

The Mood-Based Playlist Generator demonstrates the potential of combining affective computing and music recommendation systems to create personalized and emotionally resonant user experiences. By leveraging simple yet effective techniques, the application provides a lightweight and accessible solution for mood-based music recommendations. Future work could focus on enhancing the mood classification system, integrating additional music sources, and conducting user studies to evaluate the application's impact on emotional well-being.

9. References

1. Picard, R. W. (1997). *Affective Computing*. MIT Press.
2. Yang, X., Chen, L., & Chen, J. (2018). "Music Recommendation Based on Collaborative Filtering and Sentiment Analysis." *IEEE Transactions on Multimedia*, 20(12), 3321-3332.
3. Cambria, E., Hussain, A., & Havasi, C. (2020). "Sentic Computing: Techniques, Tools, and Applications." *Springer*.